Theoretical Computer Science 661 (2017) 35–55

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

A notion of effectiveness for subshifts on finitely generated groups

Nathalie Aubrun [a], Sebastián Barbieri [a,*], Mathieu Sablik [b]

[a] LIP, ENS de Lyon, CNRS, INRIA, UCBL, Université de Lyon, France
[b] Institut de Mathématiques de Toulouse, UMR5219, Université de Toulouse, CNRS, Université Paul Sabatier, France

ARTICLE INFO

Article history:
Received 11 August 2016
Accepted 25 November 2016
Available online 1 December 2016
Communicated by D. Perrin

Keywords:
Symbolic dynamics
Turing machines
Word problems
Models of computation

ABSTRACT

We generalize the classical definition of effectively closed subshift to finitely generated groups. We study classical stability properties of this class and then extend this notion by allowing the usage of an oracle to the word problem of a group. This new class of subshifts forms a conjugacy class that contains all sofic subshifts. Motivated by the question of whether there exists a group where the class of sofic subshifts coincides with that of effective subshifts, we show that the inclusion is strict for several groups, including recursively presented groups with undecidable word problem, amenable groups and groups with more than two ends. We also provide an extended model of Turing machine which uses the group itself as a tape and characterizes our extended notion of effectiveness. As applications of these machines we prove that the origin constrained domino problem is undecidable for any group of the form $G \times \mathbb{Z}$ subject to a technical condition on G and we present a simulation theorem which is valid in any finitely generated group.

0. Introduction

Symbolic dynamics were originally defined on $\mathbb{Z}$ in the highly influential article of Morse and Hedlund [14] in order to study discretization of dynamical systems. The main object in this theory is the subshift, that is, a set of colorings of a group by a finite alphabet which is defined by a set of forbidden patterns. In the case of the group $\mathbb{Z}^d$ with $d \geq 2$, it turns out that subshifts enjoy interesting computational properties, among which is the undecidability of the domino problem [5,22]. Said otherwise, there is no general algorithm deciding if there exists a coloring which avoids a finite set of patterns. This problem can be naturally generalized to any finitely generated group, nevertheless no characterization of the groups where the domino problem is undecidable is yet known, even if some partial results have arisen [21,2,4].

More recently, the use of computability theory has become essential in the study of subshifts of finite type (SFT), those defined by a finite set of forbidden patterns. For example, in $\mathbb{Z}^d$ for $d \geq 2$ the possible entropies of SFTs are characterized as right recursively enumerable numbers [16]. This type of results comes from the possibility to encode Turing machines inside $\mathbb{Z}^d$-SFTs. The study of such results led to the introduction of the class of effectively closed $\mathbb{Z}^d$-subshifts, defined by a recursively enumerable set of forbidden patterns. This class was introduced by Hochman [15] who showed that they admit an almost trivial isometric extension which is a subaction of a $\mathbb{Z}^{d+2}$-SFT. The construction was improved with two different techniques [3,11] to get a realization in sofic $\mathbb{Z}^{d+1}$-subshifts as projective subdynamics. Thus with an increase of one of the

[*] Corresponding author.
E-mail addresses: nathalie.aubrun@ens-lyon.fr (N. Aubrun), sebastian.barbieri@ens-lyon.fr (S. Barbieri), mathieu.sablik@math.univ-toulouse.fr (M. Sablik).

http://dx.doi.org/10.1016/j.tcs.2016.11.033
0304-3975/© 2016 Elsevier B.V. All rights reserved.

dimension, effectively closed $\mathbb{Z}^d$-subshifts are very close to sofic subshifts. Hochman's result suggests that if we play with the structure on which subshifts are defined, some strong links between sofic and effectively closed subshifts may emerge.

In this direction we investigate subshifts defined on infinite finitely generated groups and define a generalized notion of effectiveness. The difficulty for this task relies on the possibility, even for a finitely presented group, to have an undecidable word problem [19,6] – no algorithm can decide whether a word on the generators and their inverses represents the identity element. We study the restrictions of this class with respect to this problem and define an extended definition of effectiveness by allowing the usage of oracles to the word problem of the group.

The paper is organized as follows. Section 1 presents notations and basic notions from group theory and symbolic dynamics on finitely generated groups. In Section 2 we introduce a general model for effectively closed subshifts based on pattern codings and study its properties. We show that this class can be defined either by recursively enumerable or decidable sets of pattern codings, that it contains all subshifts of finite type and that it is stable under finite intersections. We also show that under the assumption that the underlying group is recursively presented this class can be defined using a maximal set of pattern codings, it is stable under factors, finite unions and projective subdynamics. Therefore showing that this class contains all sofic subshifts and that the property of being effectively closed is invariant under conjugacy. In order to express the limitations of this class even when the group is recursively presented we introduce in Subsection 2.2 the one-or-less-subshift $X_{\leq 1}$ which has the property of being effectively closed in recursively presented groups if and only if the word problem is decidable. This example, besides illustrating the limitations of the notion of effectively closed subshifts, answers an open question posed by Dahmani and Yaman [10,9,24]. In Subsection 2.3 we briefly introduce G-effectively closed subshifts as objects which are defined by a Turing machine with access to an oracle of the word problem of the group and we proceed to list their properties. We also show that while this is a good theoretical frame in many aspects, it does not behave well with respect to projective subactions. We end Section 2 by studying the following question: Is there a group G where the class of effectively closed subshifts coincides with the class of sofic subshifts? This question is motivated by the novel work in [1] where they show that this property is held for structures resembling subshifts defined in shears of the Baumslag–Solitar group $BS(1,2)$ under the assumption of a technical property. While their result is certainly quite specific, it raises the previous question in a natural way. We give a negative answer to that question for three classes of groups, namely:

- recursively presented groups with undecidable word problem – Theorem 2.14,
- infinite amenable groups – Theorem 2.16,
- groups which have two or more ends – Theorem 2.17.

In Section 3 we introduce an abstract model of Turing machines which instead of a bi-infinite tape uses a group. These objects are quite similar to Turing machines except that they move using a finite set of generators of G and work over patterns instead of words. This object allows us to define recursively enumerable and decidable sets of patterns and gives a way to construct explicitly Turing machines with oracles. In Theorems 3.3 and 3.4 we make this relationship explicit with the aims of concluding in Corollary 3.5 that these machines give an alternative definition of G-effectively closed subshifts by G-machines. We end Section 3 by giving two applications of these objects: In Theorem 3.7 we show that if a group G satisfies that $X_{\leq 1}$ is sofic then the origin constrained domino problem of $G \times \mathbb{Z}$ is undecidable. We also show that this implies that the domino problem for $(G \times \mathbb{Z}) * H$ is undecidable for any non-trivial finite group H. In Theorem 3.8 we show that for every infinite and finitely generated group G there exists a universal subshift U defined over $G \times \mathbb{Z}$ such that the product of U with a $G \times \mathbb{Z}$-full shift can be restricted by a finite amount of forbidden patterns and a factor code to obtain any G-effectively closed subshift as a projective subdynamics.

1. Preliminaries and notation

We assume from the reader basic knowledge about group theory and group presentations, two good references are [7,18]. For a group G we denote by 1_G its identity element. In this article we consider only finitely generated groups, and we denote by $S \subset G$ an arbitrary finite set of generators which is closed by inverses and contains the identity. If two words w_1, w_2 in S^* represent the same element in G we write $w_1 =_G w_2$. The undirected right *Cayley graph* of G given by S, denoted by $\Gamma(G,S)$, is a vertex transitive graph whose vertices are elements of G and $\{g,h\} \in \binom{G}{2}$ form an edge if there is $s \in S$ such that $gs = h$. For $g \in G$ we denote by $|g|$ the length of the shortest path from 1_G to g in $\Gamma(G,S)$. We also denote the *ball of size* $n \geq 0$ in $\Gamma(G,S)$ as $B_n = \{g \in G \mid |g| \leq n\}$. Naturally, the definitions above depend on the choice of generating set S, nevertheless all the metrics generated by the distances in such a Cayley graph are equivalent.

The *word problem* of G is defined as the formal language:

$$\mathtt{WP}(G) = \left\{ w \in S^* \mid w =_G 1_G \right\}.$$

It can be shown that the decidability of the word problem is independent of the choice of generating set S, thus the notation $\mathtt{WP}(G)$ is appropriate. A fundamental result of Novikov [19] and Boone [6] exhibits finitely presented groups with undecidable word problem.

Let $\mathcal{A}$ be a finite alphabet. We say that the set $\mathcal{A}^G = \{x : G \to \mathcal{A}\}$ equipped with the left group action $\sigma : G \times \mathcal{A}^G \to \mathcal{A}^G$ such that $(\sigma_g(x))_h = x_{g^{-1}h}$ is a *full shift*. The elements $a \in \mathcal{A}$ and $x \in \mathcal{A}^G$ are called *symbols* and *configurations* respectively.

With the discrete topology on $\mathcal{A}$, the product topology in $\mathcal{A}^G$ is compact. This topology is generated by a clopen subbasis given by the *cylinders* $[a]_g = \{x \in \mathcal{A}^G | x_g = a \in \mathcal{A}\}$. Since G is countable, $\mathcal{A}^G$ is metrizable and an ultrametric which generates the product topology is given by $d(x, y) = 2^{-\inf\{|g| \mid g \in G:\, x_g \neq y_g\}}$. A *support* is a finite subset $F \subset G$. A *pattern with support* F is an element p of $\mathcal{A}^F$. We denote the set of finite patterns by $\mathcal{A}^*_G := \bigcup_{F \subset G, |F| < \infty} \mathcal{A}^F$. For $p \in \mathcal{A}^F$ and $g \in G$ the *cylinder generated by* p *on* g is $[p]_g := \bigcap_{h \in F} [p_h]_{gh}$.

Definition. A subset X of $\mathcal{A}^G$ is a *subshift* if it is σ-invariant – $\sigma(X) \subset X$ – and closed for the cylinder topology. Equivalently, X is a subshift if there exists $\mathcal{F} \subset \mathcal{A}^*_G$ such that:

$$X = X_{\mathcal{F}} =: \bigcap_{p \in \mathcal{F}, g \in G} \mathcal{A}^G \setminus [p]_g.$$

Let X, Y be two subshifts over alphabets $\mathcal{A}_X, \mathcal{A}_Y$. We call a continuous G-equivariant – i.e. σ-commuting – function $\phi : X \to Y$ a *morphism*. A famous theorem by Curtis, Lyndon and Hedlund – see for example [7] – gives a combinatorial characterization of morphisms as block codes: namely, ϕ is a morphism if and only if there exists a finite $F \subset G$ and a local function $\Phi : \mathcal{A}^F_X \to \mathcal{A}_Y$ such that $\phi(x)_g := \Phi(\sigma_{g^{-1}}(x)|_F)$. We say ϕ is a *factor* if ϕ is surjective, and a *conjugacy* if it is bijective. Whenever there is a factor code $\phi : X \to Y$ we write $X \twoheadrightarrow Y$ and say that Y is a *factor* of X and that X is an *extension* of Y. Furthermore, if ϕ is a conjugacy we will write $X \simeq Y$ and say they are *conjugated*. The conjugacy is an equivalence relation which preserves most of the topological dynamics of a system.

We say that a subshift $X \subset \mathcal{A}^G$ is of *finite type* – SFT for short – if it can be defined by a finite set of forbidden patterns, that is, $|\mathcal{F}| < \infty$ and $X = X_{\mathcal{F}}$. We say that X is *sofic* if there exists an SFT Y and a factor code $\phi : Y \twoheadrightarrow X$. The class of sofic subshifts is the smallest class closed under factor codes that contains every SFT. Both classes are invariant under conjugacy, that is, the property of belonging to them is preserved under conjugacy.

2. Effectiveness on finitely generated groups

When $G = \mathbb{Z}$, patterns can be identified as words over a finite alphabet. We say a subshift $X \subset \mathcal{A}^{\mathbb{Z}}$ is *effectively closed* if there is a recursively enumerable set of forbidden words that defines it. We intend to generalize this definition to the class of finitely generated groups. On $\mathbb{Z}^d$, a finite pattern is no longer a word, but it can be easily coded as a word – via any recursive bijection between $\mathbb{Z}^d$ and $\mathbb{Z}$. Then effective $\mathbb{Z}^d$-subshifts correspond to subshifts which can be defined by a set of forbidden patterns that admits a recognizable set of codings. In groups with undecidable word problem this recursive bijection does not exist.

In this section we first take the previous ideas of codings to the context of finitely generated groups by introducing the formalism of pattern codings and explore the limitations of this concept when the word problem of the group is not decidable or recursively enumerable. At this point we introduce the subshift $X_{\leq 1}$ which consists in all configurations containing at most one appearance of a non-zero symbol, and use it to exemplify these previous constraints. Next we extend the notion of effectiveness by adding the power of an oracle to $\texttt{WP}(G)$. We remark the stability properties for this extended class and compare them with sofic subshifts and SFTs. Finally we exhibit three big classes of groups where this class does not coincide with the one of sofic subshifts.

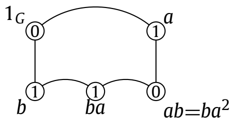
2.1. Classical effectiveness

Let G be a finitely generated group and $\mathcal{A}$ an alphabet. A *pattern coding* c is a finite set of tuples $c = (w_i, a_i)_{i \in I}$ where $w_i \in S^*$ and $a_i \in \mathcal{A}$. We say that a pattern coding is *consistent* if for every pair of tuples such that $w_i =_G w_j$ then $a_i = a_j$. For a consistent pattern coding c we define the pattern $\mathrm{p}(c) \in \mathcal{A}^F$ where $F = \bigcup_{i \in I} w_i$ and $\mathrm{p}(c)_{w_i} = a_i$.

Example. Let $BS(1,2) \cong \langle a, b \mid ab = ba^2 \rangle$ be a Baumslag–Solitar group and $\mathcal{A} = \{0, 1\}$. Then the pattern coding

$$\begin{array}{ccc} (\epsilon, 0) & (b, 1) & (a, 1) \\ (ab, 0) & (ba^2, 0) & (ba, 1) \end{array}$$

is consistent, since all the words above on $S = \{a, b, a^{-1}, b^{-1}\}$ represent different elements in G except for ab and ba^2 that are assigned the same symbol. The pattern it defines is:



But the pattern coding

$$\begin{array}{lll} (\epsilon, 0) & (a^2, 1) & (bab^{-1}a, 1) \\ (a, 1) & (ba, 1) & (abab^{-1}, 0) \end{array}$$

is inconsistent since words $abab^{-1}$ and $bab^{-1}a$ represent the same element in G but are assigned different symbols.

A set of pattern codings $\mathcal{C}$ is said to be recursively enumerable if there is a Turing machine which takes as input a pattern coding c and accepts it if and only if $c \in \mathcal{C}$.

Definition. A subshift $X \subset \mathcal{A}^G$ is *effectively closed* if there is a recursively enumerable set of pattern codings $\mathcal{C}$ such that:

$$X = X_{\mathcal{C}} := \bigcap_{g \in G, c \in \mathcal{C}} \left(\mathcal{A}^G \setminus \bigcap_{(w,a) \in c} [a]_{gw} \right).$$

The specific choice of the set of generators S is irrelevant as one can easily translate one in terms of the other. Notice that if a pattern coding is inconsistent then $\bigcap_{(w,a)\in c}[a]_w = \emptyset$ and that if it is consistent then $\bigcap_{(w,a)\in c}[a]_w = [\mathrm{p}(c)]_{1_G}$. Therefore, the subshift defined by a set of pattern codings $\mathcal{C}$ only depends on the set of consistent ones, in the sense that if $\mathrm{p}(\mathcal{C})$ is the set of patterns defined by the consistent pattern codings of $\mathcal{C}$ then $X_{\mathcal{C}} = X_{\mathrm{p}(\mathcal{C})}$.

We could also define this class by the existence of a decidable family rather than a recursively enumerable one. This justifies the usage of the word "effectively". The following proposition is commonly known to hold true in $G = \mathbb{Z}^d$. Here we present a general version which works in every finitely generated group.

Proposition 2.1. *Let $X \subset \mathcal{A}^G$ be an effectively closed subshift. Then there exists a decidable set of pattern codings $\mathcal{C}$ such that $X = X_{\mathcal{C}}$.*

Proof. Let $\mathcal{C}'$ be a recursively enumerable set of pattern codings such that $X = X_{\mathcal{C}'}$. If $\mathcal{C}'$ is finite the result is trivial. Otherwise there exists a recursive enumeration $\mathcal{C}' = \{c_0, c_1, \dots\}$. For a pattern coding c we define its length as $|c| = \max_{(w,a)\in c} |w|$. For $n \in \mathbb{N}$ let $L_n = \max_{k \le n} |c_k|$ and define $\mathcal{C}_n$ as the finite set of all pattern codings c which satisfy the following properties:

- Every $w \in S^*$ with $|w| \le L_n$ appears in exactly one pair in c.
- $(w, a) \in c$ implies that $|w| \le L_n$.
- If $(w, a) \in c_n$ then $(w, a) \in c$.

That is, $\mathcal{C}_n$ is the set of all pattern codings which are completions of c_n up to every word of length at most L_n in every possible way. Consider $\mathcal{C} = \bigcup_{n\in\mathbb{N}} \mathcal{C}_n$. Clearly it satisfies that $X = X_{\mathcal{C}}$. We claim it is decidable.

Consider the algorithm which does the following on input c: It initializes n to 0. Then it enters into the following loop: First it produces the pattern coding c_n. If $L_n > |c|$ it rejects the input. Otherwise it calculates the set $\mathcal{C}_n$. If $c \in \mathcal{C}_n$ then it accepts, otherwise it increases the value of n by 1.

As L_n is increasing and cannot stay in the same value indefinitely this algorithm eventually ends for every input. □

In what follows we will show which are the liberties one can take when choosing a defining set of pattern codings and the structural properties of this class. Some of these are related to the following notion in group theory. A group is said to be *recursively presented* if there is a presentation $G \cong \langle S, R \rangle$ where S is a recursive set and $R \subset S^*$ a recursively enumerable language. As we only consider finitely generated groups S is always finite and thus recursive, so we take the second requirement as the definition.

Proposition 2.2. *Let G be a finitely generated group and $\mathcal{A}$ be an alphabet with at least two symbols. The following are equivalent:*

1. *G is recursively presented.*
2. *$\mathtt{WP}(G)$ is recursively enumerable.*
3. *The set of inconsistent pattern codings is recursively enumerable.*

Proof. The equivalence between the two first statements is trivial. Let G have recursively enumerable word problem. As $u =_G v \Leftrightarrow uv^{-1} =_G 1_G$ the set of inconsistent pattern codings is recursively enumerable. Indeed, for $n \in \mathbb{N}$, a Turing machine on input c can simulate iteratively for n steps the machine recognizing $\mathtt{WP}(G)$ applied to uv^{-1} for every pair $(u, a), (v, b) \in c$ with $a \ne b \in \mathcal{A}$ and accept if this procedure accepts for some n. Conversely, given $w \in S^*$, it suffices to give as input to the machine recognizing the inconsistency of the pattern codings $c = \{(\epsilon, a), (w, b)\}$ with $a \ne b \in \mathcal{A}$ in order to recognize if $w =_G 1_G$. □

Lemma 2.3. *Let $X \subset \mathcal{A}^G$ be an effectively closed subshift. If G is recursively presented then it is possible to choose $\mathcal{C}$ to be a recursively enumerable and maximal – for inclusion – set of pattern codings such that $X = X_{\mathcal{C}}$.*

This lemma is fundamental in the rest of the article. Indeed, every time the statement of a result requires as hypothesis that a group G is recursively presented, this is because its proof uses the existence of a recursively enumerable and maximal set of pattern codings for some G-subshift.

Proof. A pattern coding c belongs to the maximal set $\mathcal{C}$ defining X if and only if $X \cap \bigcap_{(w,a)\in c}[a]_w = \emptyset$. Let $c \in \mathcal{C}$ and $\mathcal{C}'$ be a recursively enumerable set such that $X = X_{\mathcal{C}'}$. Then:

$$\bigcap_{(w,a)\in c}[a]_w \subset \bigcup_{c'\in\mathcal{C}', g\in G} \bigcap_{(w',a')\in c'} [a']_{gw'}.$$

By compactness we may extract a finite open cover indexed by c'_i, g_i such that:

$$\bigcap_{(w,a)\in c}[a]_w \subset \bigcup_{i\le n} \bigcap_{(w',a')\in c'_i} [a']_{g_i w'} \tag{1}$$

Note that each of these g_i can be seen as a finite word in S^*. Now let T be the Turing machine which does iteratively for $n \in \mathbb{N}$ the following:

- Runs n steps the machine T_1 recognizing $\mathtt{WP}(G)$ for every word in S^* of length smaller than n.
- Runs n steps the machine T_2 recognizing $\mathcal{C}'$ for every pattern coding defined on a subset of words of S^* of length smaller than n.
- Let $\sim_n$ be the equivalence relation for words in S^* of length smaller than n such that $u \sim_n v$ if uv^{-1} has been already accepted by T_1. Let $\mathcal{C}_n$ be the pattern codings already accepted by T_2. If every word in c has length smaller than n check if the following relation is true under $\sim_n$:

$$\bigcap_{(w,a)\in c}[a]_w \subset \bigcup_{c'\in\mathcal{C}_n, |u|\le n} \bigcap_{(w',a')\in c'} [a']_{uw'}$$

  If it is true, accept, otherwise increase n by 1 and continue.

Let m be the max of all $|w|$ such that $(w,a) \in c$, and $|w'|$ such that $(w',a') \in c'_i$ and all $|g_i|$. By definition, there exists an $N \in \mathbb{N}$ such that every c'_i for $i \le n$ is accepted and every word representing 1_G of length smaller than $2m$ is accepted. This means that at stage N relation (1) is satisfied and T accepts c. If c is not in the maximal set, the machine never accepts. □

Lemma 2.3 is no longer true if G is not recursively presented. Indeed, the maximal set of pattern codings defining the full shift is given by the set of all inconsistent pattern codings, which is recursively enumerable if and only if G is recursively presented by Proposition 2.2.

Proposition 2.4. *The class of SFTs is contained in the class of effectively closed subshifts.*

Proof. Let X be an SFT. Then $X = X_{\mathcal{F}}$ for a finite set $\mathcal{F}$. For each $p \in \mathcal{F}$ consider a pattern coding c_p such that $\mathrm{p}(c_p) = p$ and let $\mathcal{C} = \{c_p \mid p \in \mathcal{F}\}$. Clearly $X = X_{\mathcal{C}}$ and as $\mathcal{C}$ is finite it is recursively enumerable. □

Proposition 2.5. *The class of effectively closed subshifts is closed by finite intersections.*

Proof. Let $X = X_{\mathcal{C}_X}$ and $Y = Y_{\mathcal{C}_Y}$ be effectively closed subshifts. Without loss of generality suppose $X, Y \subset \mathcal{A}^G$ (same alphabet) and note that:

$$\begin{aligned} X \cap Y &= \left(\mathcal{A}^G \setminus \bigcup_{g\in G, c\in\mathcal{C}_X} \bigcap_{(w,a)\in c} [a]_{gw}\right) \cap \left(\mathcal{A}^G \setminus \bigcup_{g\in G, c\in\mathcal{C}_Y} \bigcap_{(w,a)\in c} [a]_{gw}\right) \\ &= \mathcal{A}^G \setminus \bigcup_{g\in G, c\in\mathcal{C}_X\cup\mathcal{C}_Y} \bigcap_{(w,a)\in c} [a]_{gw} \\ &= X_{\mathcal{C}_X\cup\mathcal{C}_Y} \end{aligned}$$

Therefore, it suffices on input c to launch the Turing machines recognizing $\mathcal{C}_X$ and $\mathcal{C}_Y$ in parallel and accept if either of them accepts. □

The result obviously does not extend to countable intersections. If it were so, since every possible subshift is obtainable as an intersection of SFTs (enumerate the forbidden patterns, define $X_n = X_{p_1,\dots,p_n}$, then $X = \bigcap_{n\in\mathbb{N}} X_n$), we would conclude that all subshifts are effectively closed. But there is an uncountable number of subshifts on a fixed alphabet, and effectively closed subshifts clearly constitute a countable set, so there must be one that is not effectively closed.

Proposition 2.6. *For a recursively presented group the class of effectively closed subshifts is closed by finite unions.*

Proof. Let $X = X_{\mathcal{C}_X}$ and $Y = Y_{\mathcal{C}_Y}$ be effectively closed subshifts. As G is recursively presented we can suppose $\mathcal{C}_X$ and $\mathcal{C}_Y$ are maximal as in Lemma 2.3. As in the previous proof we can show:

$$X \cup Y = \mathcal{A}^G \setminus \left(\left(\bigcup_{g\in G, c\in\mathcal{C}_X} \bigcap_{(w,a)\in c} [a]_{gw} \right) \cap \left(\bigcup_{g\in G, c\in\mathcal{C}_Y} \bigcap_{(w,a)\in c} [a]_{gw} \right) \right)$$

Thus, as these sets are maximal we have $X \cup Y = X_{\mathcal{C}_X \cap \mathcal{C}_Y}$. It suffices therefore to launch both Turing machines and accept if both accept. □

Proposition 2.7. *For recursively presented groups the class of effectively closed subshifts is closed under factors.*

Proof. Let $X \subset \mathcal{A}_X^G$ be an effectively closed subshift. As G is recursively presented, the recursively enumerable set of pattern codings $\mathcal{C}_X$ can be chosen to be maximal by Lemma 2.3. Consider a factor code $\phi : X \twoheadrightarrow Y$ defined by a local function $\Phi : \mathcal{A}_X^F \to \mathcal{A}_Y$ with $F = \{f_1, \dots, f_{|F|}\}$ where $f_1, \dots, f_{|F|}$ are words of S^*.

As ϕ is surjective, for each $a \in \mathcal{A}_Y$ then $|\Phi^{-1}(a)| > 0$. Therefore we can associate to a pair (w, a) a non-empty finite set of pattern codings

$$\mathcal{C}_{w,a} = \{(wf_i, p_{f_i})_{i=1,\dots,|F|} \mid p \in \Phi^{-1}(a)\}.$$

That is, $\mathcal{C}_{w,a}$ is a finite set of pattern codings over $\mathcal{A}_X$ representing every possible preimage of a. For a pattern coding $c = (w_i, a_i)_{i\leq n}$ where $a_i \in \mathcal{A}_Y$ we define:

$$\mathcal{C}_c = \{ \bigcup_{(w,a)\in c} \widetilde{c}_{w,a} \mid \widetilde{c}_{w,a} \in \mathcal{C}_{w,a}\}.$$

That is, $\mathcal{C}_c$ is the finite set of pattern codings formed by choosing one possible preimage for each letter. This set has the property that if Φ is applied pointwise then $\Phi(\mathrm{p}(\mathcal{C}_c)) = \{\mathrm{p}(c)\}$. Let T be the Turing machine which on input c runs the machine recognizing $\mathcal{C}_X$ on every pattern coding in $\mathcal{C}_c$. If it accepts for every input, then T accepts c. Let $\mathcal{C}_Y$ be the set of pattern codings accepted by T. We claim $Y = Y_{\mathcal{C}_Y}$.

Let $y \in Y_{\mathcal{C}_Y}$ and $n \in \mathbb{N}$. For each pattern coding c such that $\mathrm{p}(c) = y|_{B_n}$, there is a pattern coding $c_n \in \mathcal{C}_c$ which does not belong to $\mathcal{C}_X$. As $\mathcal{C}_X$ is maximal we have that $[\mathrm{p}(c_n)] \cap X \neq \emptyset$. Extracting a configuration x_n from $[\mathrm{p}(c_n)] \cap X$ we obtain a sequence $(x_n)_{n\in\mathbb{N}}$. By compactness there is a converging subsequence with limit $\widetilde{x} \in X$. By continuity of ϕ we have that $y = \phi(\widetilde{x}) \in Y$. Conversely if $y \in Y$ there exists $x \in X$ such that $\phi(x) = y$. Therefore for every finite $F' \subset G$ and pattern coding c with $\mathrm{p}(c) = y|_{F'}$ there exists a pattern coding $\widetilde{c} \in \mathcal{C}_c$ such that $\mathrm{p}(\widetilde{c}) = x|_{F'F}$. Therefore, $c \notin \mathcal{C}_y$ and thus $y \in Y_{\mathcal{C}_Y}$. □

Corollary 2.8. *For a recursively presented group the following are true:*

- *The class of effectively closed subshifts is invariant under conjugacy.*
- *The class of effectively closed subshifts contains all sofic subshifts.*

We do not know if the previous results extend to the general case where G is not recursively presented. The main obstruction is that without that hypothesis there is no control on the representations of the finite set F which defines the local rule of the factor. As an example, suppose $F = \{1_G\}$, that is $\Phi : \mathcal{A}_X \to \mathcal{A}_Y$. In order to detect forbidden patterns by using the recursively enumerable set defining X we would need to touch all possible representations of F, which is exactly the set $\mathtt{WP}(G)$.

Let $H \leq G$ be a subgroup of G. Given a subshift $X \subset \mathcal{A}^G$ the H-projective subdynamics of X is the subshift $\pi_H(X) \subset \mathcal{A}^H$ defined as:

$$\pi_H(X) = \{x \in \mathcal{A}^H \mid \exists y \in X, \forall h \in H, x_h = y_h\}$$

Proposition 2.9. *Let G be a recursively presented group and $H \leq G$ a finitely generated subgroup of G. If $X \subset \mathcal{A}^G$ is effectively closed, then its H-projective subdynamics $\pi_H(X)$ is effectively closed.*

Proof. As H is finitely generated, there exists a finite set $S' \subset H$ such that $\langle S' \rangle = H$. As G is finitely generated by S there exists a function $\gamma : S' \to S^*$ such that $s' =_G \gamma(s')$ (that is, every element of S' can be written as a word in S^*). Extend the function γ to act by concatenation over words in S'^*.

As G is recursively presented, by Lemma 2.3 the set of pattern codings $\mathcal{C}_G$ defining X can be chosen to be maximal. Let $c = (w_i, a_i)_{i \in I}$ a pattern coding where $w_i \in S'^*$ and consider $\gamma(c) = (\gamma(w_i), a_i)_{i \in I}$. Let T be the Turing machine which on input c runs the algorithm recognizing $\mathcal{C}_G$ on input $\gamma(c)$ and accepts if and only if this machine accepts. Clearly $\mathcal{C}_H = \{c \mid T \text{ accepts } c\}$ is recursively enumerable. Also, as $\mathcal{C}_G$ is a maximal set of pattern codings then $c \in \mathcal{C}_H \iff [\mathrm{p}(\gamma(c))] \cap X = \emptyset$. Therefore $\pi_H(X) = X_{\mathcal{C}_H}$. □

Besides all of these obstructions, even for recursively presented groups there are very simple subshifts which do not fall in this class. In order to illustrate this limitation we introduce the *One-or-less subshift*.

2.2. The one-or-less subshift

Consider the subshift $X_{\leq 1} \subset \{0,1\}^G$ whose configurations contain at most one appearance of the letter 1.

$$X_{\leq 1} = \{x \in \{0,1\}^G \mid 1 \in \{x_g, x_h\} \implies g = h\}$$

As we shall see later, this subshift is related to the word problem of a group. In the literature, it is sometimes called the "sunny side up" subshift. We begin by showing some properties of $X_{\leq 1}$.

Proposition 2.10. *If G is infinite, then $X_{\leq 1}$ is not an SFT.*

Proof. Suppose $X_{\leq 1} = X_{\mathcal{F}}$ for a finite $\mathcal{F}$ and let $F = \bigcup_{p \in \mathcal{F}} \mathrm{supp}(p)$, $U = \bigcup_{h \in F^{-1}} hF$ and note that $|U| < \infty$. As G is infinite, there exists $g \in G \setminus U$. Consider the configuration $x \in \{0,1\}^G$ which takes the value 1 in $\{1_G, g\}$ and 0 elsewhere. Clearly $x \notin [p]_h$ for every $h \in G$ and $p \in \mathcal{F}$ otherwise $\{1_G, g\} \subset hF$ implying that $hF \subset U$ and thus $g \in U$. Therefore $x \in X_{\mathcal{F}}$ but $x \notin X_{\leq 1}$. □

This subshift has already been studied in [10]. In that article the authors showed that the action of a relatively hyperbolic group on its boundary is related to $X_{\leq 1}$ being sofic. They said a group G has the *special symbol property* if $X_{\leq 1} \subset \{0,1\}^G$ is a sofic subshift. They furthermore proved some stability properties among which are:

1. If G has the special symbol property then G is finitely generated.
2. If G splits in a short exact sequence $1 \to N \to G \to H \to 1$ and both N and H satisfy the special symbol property, then G also does.
3. If $[G : H] < \infty$ then G has the special symbol property if and only if H does.
4. The special symbol property is true for:
   - Finitely generated free groups.
   - Finitely generated abelian groups.
   - Hyperbolic groups.
   - Poly-hyperbolic groups.

Besides the restriction of G being finitely generated the authors did not present any example of group without the special symbol property. In this section we introduce a computability obstruction for this property which at the same time shows one of the limitations of the classical approach to effectiveness.

Proposition 2.11. *Let G be a recursively presented group. Then $X_{\leq 1}$ is effectively closed if and only if $\mathtt{WP}(G)$ is decidable.*

Proof. If $\mathtt{WP}(G)$ is decidable then $X_{\leq 1}$ is effectively closed. Indeed, an algorithm recognizing a maximal set of pattern codings $\mathcal{C}$ such that $X_{\leq 1} = X_{\mathcal{C}}$ is the following: On input c it considers every pair $(w_1, 1), (w_2, 1)$ in c and accepts if and only if $w_1 w_2^{-1} \neq_G 1_G$ for a pair. Conversely, as G is recursively presented, the word problem is already recursively enumerable. It suffices to show it is co-recursively enumerable.

By Lemma 2.3 there exists a maximal set of forbidden pattern codings $\mathcal{C}$ with $X_{\leq 1} = X_{\mathcal{C}}$. Given $w \in S^*$, consider the pattern coding $c_w = \{(\epsilon, 1), (w, 1)\}$. Note that $w \neq_G 1_G \iff c_w \in \mathcal{C}$. The algorithm which on input $w \in S^*$ runs the algorithm recognizing $\mathcal{C}$ on input c_w and accepts if and only if this one accepts recognizes $S^* \setminus \mathtt{WP}(G)$. Hence $\mathtt{WP}(G)$ is co-recursively enumerable. □

Using Proposition 2.7 we obtain the following corollary which answers a question of Dahmani and Yaman [9,24].

Corollary 2.12. *If G is recursively presented and $\mathtt{WP}(G)$ is undecidable, then $X_{\leq 1}$ is not sofic.*

2.3. G-Effectiveness

In order to escape the limitations of Lemma 2.3 and Proposition 2.7 and include subshifts such as $X_{\leq 1}$, we introduce the class of G-effectively closed subshifts. The aim of this subsection is to briefly introduce these objects and remark their properties. They are studied in detail in Section 3.

A set of pattern codings $\mathcal{C}$ is said to be *recursively enumerable with oracle* $\mathcal{O}$ if there exists a Turing machine with oracle $\mathcal{O}$ which accepts input c if and only if $c \in \mathcal{C}$. The oracle $\mathcal{O}$ is a language to which these special machines have the right to ask if $w \in \mathcal{O}$ and receive the correct answer in one step.

Definition. A subshift $X \subset \mathcal{A}^G$ is *G-effectively closed* if there is a set of pattern codings $\mathcal{C}$ such that $X = X_{\mathcal{C}}$, and $\mathcal{C}$ is recursively enumerable with oracle $\mathtt{WP}(G)$.

We remark the following properties that either fall directly from the definition or are obtained from adding the word problem $\mathtt{WP}(G)$ as oracle to the previous results. Let G be a finitely generated group, then:

1. If X is a G-effectively closed subshift then a maximal set of pattern codings $\mathcal{C}$ such that $X = X_{\mathcal{C}}$ is recursively enumerable with oracle $\mathtt{WP}(G)$.
2. The class of G-effectively closed subshifts is closed under finite intersections and unions.
3. The class of G-effectively closed subshifts is closed under factors.
4. Being G-effectively closed is a conjugacy invariant.
5. The class of G-effectively closed subshifts contains all sofic subshifts.
6. The class of G-effectively closed subshifts contains all effectively closed subshifts.
7. If $\mathtt{WP}(G)$ is decidable, then every G-effectively closed subshift is effectively closed.
8. $X_{\leq 1}$ is a G-effectively closed subshift.

The only property which does not extend nicely is the stability under taking projective subdynamics. Clearly if $X \subset \mathcal{A}^G$ is G-effectively closed then for any finitely generated $H \leq G$ we would have that the H-projective subdynamics $\pi_H(X)$ can be defined by a set of pattern codings which is recursively enumerable with oracle $\mathtt{WP}(G)$. Nevertheless, it may not be possible to define such set with Turing machines using oracle $\mathtt{WP}(H)$.

Proposition 2.13. *Let G be a group which is not recursively presented. There exists a $(G \times \mathbb{Z})$-effectively closed subshift $X \subset \mathcal{A}^{G\times\mathbb{Z}}$ such that its $\mathbb{Z}$-projective subdynamics is not $\mathbb{Z}$-effectively closed.*

Proof. Let $\mathcal{A} = S \cup \{\star\}$. For $w \in S^*$, let p_w be defined over the support $\{1_G\} \times \{0, \dots |w| + 1\}$ such that $(p_w)_{(1_G,0)} = (p_w)_{(1_G,|w|+1)} = \star$ and for $j \in \{1, \dots, |w|\}$ then $(p_w)_{(1_G,j)} = w_j$. Let $X := X_{\mathcal{F}} \subset \mathcal{A}^{G\times\mathbb{Z}}$ be defined by the set of forbidden patterns $\mathcal{F} = \{p_w \mid w \in \mathtt{WP}(G)\}$. Clearly X is $(G \times \mathbb{Z})$-effectively closed. Every $\mathbb{Z}$-coset of a configuration $x \in X$ contains a bi-infinite sequence $y \in \mathcal{A}^{\mathbb{Z}}$ such that either y contains at most one symbol $\star$ or every word appearing between two appearances of $\star$ represents 1_G in G.

We claim that $\pi_{\mathbb{Z}}(X)$ is not effectively closed. If it were, there would exist a maximal set of forbidden pattern codings which is recursively enumerable and defines $\pi_{\mathbb{Z}}(X)$. Therefore given $w \in S^*$ a machine could run the algorithm for the word $\star w \star$ and it would be accepted if and only if $w =_G 1_G$. This would imply that G is recursively presented. □

In Section 3 a characterization of these subshifts by Turing machines which instead of a tape have Cayley graphs of groups is given. This allows an alternative definition of G-effectiveness which at the same time gives a concrete construction of Turing machines with oracle.

2.4. Groups with G-effective subshifts which are not sofic

In the work of two of the authors [1], it is shown that for subshifts in the hyperbolic plane that satisfy a technical condition, the property of being sofic is equal to the property of being effectively closed. By hyperbolic plane it is meant the monoid $M = \langle a, a^{-1}, b \mid ab = ba^2, aa^{-1} = 1_M\rangle$ which looks like a shear of the Baumslag–Solitar group $BS(1, 2)$ (here all the definitions given above for groups naturally extend to monoids). The reason behind this fact is that the doubling structure of this monoid allows to transmit the information on a row $b^n\langle a\rangle$ to all rows $b^m\langle a\rangle$ where $m \geq n$, and thus a Turing machine calculation can be implemented as an extra SFT extension. This shows that any subshift defined by a recursively enumerated set of pattern codings is in fact a sofic subshift.

This result raises the following questions:

- If we consider the group $BS(1, 2)$, is it true that every effectively closed subshift is sofic?
- Is there any group G such that every G-effectively closed subshift is sofic?
- Is there any group such that the class of effectively closed subshifts and sofic subshifts coincide?

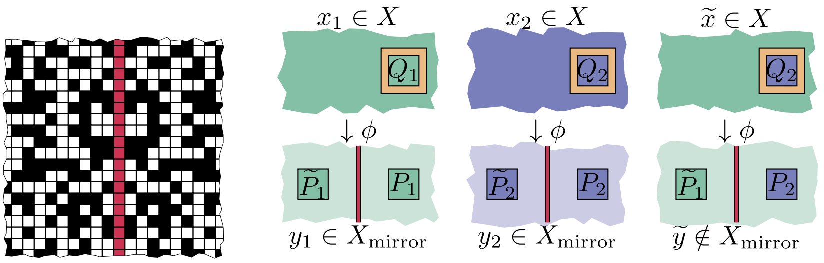


Fig. 1. Configuration in the mirror shift and technique showing non-soficity.

In this section we give a negative answer to the first question, and give partial negative answers to the second and third questions. More precisely, we show that the equality between the class of G-effectively closed subshifts and sofic subshifts cannot happen in three cases: recursively presented groups with undecidable word problem, amenable groups and groups with two or more ends.

Theorem 2.14. *For every recursively presented group G with undecidable word problem there exists a G-effectively closed subshift which is not sofic.*

Proof. The subshift $X_{\leq 1}$ is G-effectively closed but not sofic for recursively presented G as stated in Corollary 2.12. □

Clearly, this does not say anything about the existence of effectively closed subshifts which are not sofic when the word problem is undecidable. In fact, it is not even known whether $X_{\leq 1}$ is sofic for all groups with decidable word problem.

For the case of amenable groups, we take inspiration in a classical construction for $\mathbb{Z}^2$ called the mirror shift. It consists of all configurations over the alphabet $\mathcal{A} = \{$ □, ■, 🟥 $\}$ such that these forbidden patterns do not appear.

$$\mathcal{F} := \left\{ \begin{smallmatrix}□\\🟥\end{smallmatrix}, \begin{smallmatrix}■\\🟥\end{smallmatrix}, \begin{smallmatrix}🟥\\□\end{smallmatrix}, \begin{smallmatrix}🟥\\■\end{smallmatrix} \right\} \cup \bigcup_{w \in \mathcal{A}^*} \left\{ 🟥\, w\, 🟥,\ ■\, w\, 🟥\, \tilde{w}\, □,\ □\, w\, 🟥\, \tilde{w}\, ■ \right\},$$

where $\tilde{w}$ denotes the reverse of the word w.

This subshift is easily seen to be effectively closed, while it can be proven that it is not sofic. Indeed, if S is the canonical set of generators of $\mathbb{Z}^2$, then $|B_{n+1} \setminus B_n|/|B_n|$ tends to 0 as n goes to infinity. From this it is possible to deduce that in a suitable SFT extension of the mirror shift, there are two different patterns sharing the same boundary which yield different patterns in the mirror subshift. As shown in Fig. 1, switching a pattern for the other produces a point outside the subshift yielding a contradiction. In what follows we generalize this technique to amenable groups.

For a finitely generated group we say a sequence of elements $(g_n)_{n \in \mathbb{N}}$ is recursive if there is a Turing machine which on input n produces a word $w \in S^*$ such that $w =_G g_n$. If the Turing machine uses oracle $\mathcal{O}$ then the sequence is said to be recursive with oracle $\mathcal{O}$.

Lemma 2.15. *For every infinite group G there exists a pair of recursive sequences $(g_n)_{n\in\mathbb{N}}$, $(h_n)_{n\in\mathbb{N}}$ with oracle $\mathtt{WP}(G)$ such that the family of sets*

$$\mathcal{S} = \{\{1_G\}\} \cup \{g_n B_n\}_{n\in\mathbb{N}} \cup \{h_n B_n\}_{n\in\mathbb{N}}$$

is pairwise disjoint.

Proof. Fix a total order on S and extend it to a lexicographic order in S^*. Let T_g, T_h be the Turing machines with oracle $\mathtt{WP}(G)$ that do the following on input $n \in \mathbb{N}$.

- Let $N = 1 + 2\sum_{k=1}^{n}(2k+1) = 1 + 2n(n+2)$. Solve the word problem for every $w \in S^*$ such that $|w| \leq 2N$. This allows to construct B_N of the Cayley graph $\Gamma(G, S)$.
- Assign the value 0 to every $g \in B_N \setminus \{1_G\}$, and 1 to 1_G. Assign initially the value $g_0, \dots, g_n, h_0, \dots, h_n$ to ϵ. And initiate a variable k with its value set initially to 0.
- While $k \leq n$ do the following: Iterate over all $w \in S^*$ lexicographically. If for w all of the values of wB_k have the value 0 then:
  - Turn all of the values in wB_k to 1.
  - if $g_k = \epsilon$ set $g_k = w$.
  - otherwise, set $h_k = w$ and assign $k \leftarrow k + 1$.
- For the machine T_g return g_n, for T_h return h_n.

As G is infinite and finitely generated there exist elements of arbitrary length. Therefore the bound N suffices to construct all these disjoint balls: Indeed, it is the sum of the diameters of the considered sets. Moreover, as the lexicographic order is fixed beforehand this algorithm will always produce the same values, therefore it gives a recursive enumeration of the desired sets. □

Theorem 2.16. *Let G be an infinite amenable group. Then there exists a G-effectively closed subshift which is not sofic.*

Proof. Let $(g_n)_{n\in\mathbb{N}}$, $(h_n)_{n\in\mathbb{N}}$ be recursive sequences with oracle $\texttt{WP}(G)$ as in Lemma 2.15, and consider the subshift $Y \subset \{0,1,2\}^G$ defined as $Y = Y_1 \cap Y_2$ where:

$$Y_1 = \{y \in \{0,1,2\}^G \mid 2 \in \{y_g, y_h\} \implies g = h\}$$

$$Y_2 = \{y \in \{0,1,2\}^G \mid y_g = 2 \implies \forall n \in \mathbb{N}, \sigma_{g_n^{-1}g^{-1}}(y)|_{B_n} = \sigma_{h_n^{-1}g^{-1}}(y)|_{B_n}\}$$

It is clear these two sets are closed and shift-invariant, thus Y is a subshift. Moreover, they are both G-effectively closed subshifts: Y_1 is defined by all pattern codings which contain a pair $(w_1,2),(w_2,2)$ such that $w_1 \neq_G w_2$ and Y_2 by all pattern codings which contain a triple $(w_1,2),(w_2,a),(w_3,b)$ with $a \neq b$ for which there exists $n \in \mathbb{N}$ and $h \in B_n$ such that $w_2 =_G w_1 g_n h$ and $w_3 =_G w_1 h_n h$. As the sequences are recursive with $\texttt{WP}(G)$ as oracle this is an effectively enumerable set with oracle $\texttt{WP}(G)$. As the class of G-effectively closed subshifts is closed under intersections we obtain that Y is G-effectively closed.

We are going to show that Y is not sofic. As G is amenable (see [7]), for each $\varepsilon > 0$ and finite $K \subset G$ there exists a non-empty finite set $F \subset G$ such that:

$$\forall k \in K, \frac{|F \setminus Fk|}{|F|} < \varepsilon$$

Suppose Y is sofic, then there exists an SFT $X \subset \mathcal{B}^G$ and a factor code $\phi : X \twoheadrightarrow Y$. Without loss of generality one can suppose that ϕ is a 1-block code, that is, it is defined by a local rule $\Phi : \mathcal{B} \to \mathcal{A}$. Indeed, if this was not the case, and $\Phi : \mathcal{B}^F \to \mathcal{A}$ for $F \neq \{1_G\}$ we can find a conjugated version of X over the alphabet $\widetilde{\mathcal{B}} := \mathcal{B}^F$ which is given by the conjugacy $\widetilde{\phi} : X \to \widetilde{X}$ such that $\widetilde{\phi}(x)_g = \sigma_{g^{-1}}(x)|_F$. As being SFT is a conjugacy invariant we can choose without loss of generality $\widetilde{X}$ as the extension.

Let K be the union of the supports of $p \in \mathcal{F}$ where $X = X_{\mathcal{F}}$ and $|\mathcal{F}| < \infty$, $\varepsilon = \frac{log(2)}{|K|log(|\mathcal{B}|)}$ and for simplicity denote $\partial_K F = F \setminus \bigcap_{k\in K} Fk$. We obtain that there is F such that:

$$\frac{|\partial_K F|}{|F|} \leq \sum_{k\in K} \frac{|F \setminus Fk|}{|F|} < |K| \frac{log(2)}{|K|log(|\mathcal{B}|)} = \frac{log(2)}{log(|\mathcal{B}|)}$$

Note that the previous property is invariant by translation, that is, if F satisfies this property, then gF also does for each $g \in G$. By choosing a large enough $n \in \mathbb{N}$ such that $F \subset B_n$, then $g_n F \subset g_n B_n$.

Putting everything together, we can find a set F such that $|\mathcal{B}|^{|\partial F|} < 2^{|F|}$ and there exists $n \in \mathbb{N}$ such that $1_G \notin g_n F$, $g_n F \subset g_n B_n$ and $g_n F \cap h_n B_n = \emptyset$.

Consider the set of patterns:

$$\mathcal{P} = \{p : \{1_G\} \cup g_n F \to \{0,1,2\} \mid p_{1_G} = 2, \forall h \in g_n F : p_h \in \{0,1\}\}$$

Clearly $|\mathcal{P}| = 2^{|F|}$. As $g_n F \subset g_n B_n$ then for each $p \in \mathcal{P}$, $[p]_{1_G} \cap Y \neq \emptyset$. Let $y^p \in [p]_{1_G} \cap Y$ and $x^p \in X$ such that $\phi(x^p) = y^p$. As $|\mathcal{B}|^{|\partial F|} < 2^{|F|}$ by pigeonhole principle there are $x^{p_1} \neq x^{p_2}$ such that $x^{p_1}|_{g_n\partial F} = x^{p_2}|_{g_n\partial F}$.

By definition of K we obtain that $\widetilde{x} \in X$ where $\widetilde{x}$ is the configuration defined as $\widetilde{x}|_F = x^{p_1}|_F$ and $\widetilde{x}|_{G\setminus F} = x^{p_2}|_{G\setminus F}$. As ϕ is a 1-block code we get that $\phi(\widetilde{x})|_F = y^{p_1}|_F$ and $\phi(\widetilde{x})|_{G\setminus F} = y^{p_2}|_{G\setminus F}$. Consider $\bar{g} \in B_n$ such that $(y^{p_1})_{g_n\bar{g}} \neq (y^{p_2})_{g_n\bar{g}}$. Then:

$$\phi(\widetilde{x})_{h_n\bar{g}} = (y^{p_2})_{h_n\bar{g}} = (y^{p_2})_{g_n\bar{g}}$$

$$\phi(\widetilde{x})_{g_n\bar{g}} = (y^{p_1})_{g_n\bar{g}}$$

Therefore $\phi(\widetilde{x})_{h_n\bar{g}} \neq \phi(\widetilde{x})_{g_n\bar{g}}$ but $\phi(\widetilde{x})_{1_G} = 2$ which means that $\phi(\widetilde{x}) \notin Y$. □

In particular, this theorem gives a negative answer in the case of $BS(1,2)$ which is solvable and thus amenable.

Definition. The *number of ends* $e(G)$ of the group G is the limit as n tends to infinity of the number of infinite connected components of $\Gamma(G,S) \setminus B_n$.

The number of ends is a quasi-isomorphism invariant and thus it does not depend on the choice of S. It is also known that for a finitely generated group G then $e(G) \in \{0,1,2,\infty\}$. Stallings theorem about ends of groups [23] gives a constructive characterization of the groups satisfying $e(G) \geq 2$. In particular we have $e(G) = 2$ if and only if G is infinite and virtually cyclic.

Theorem 2.17. *Let G be a finitely generated group where $e(G) \geq 2$. Then there are G-effectively closed subshifts which are not sofic.*

Proof. Let $N \in \mathbb{N}$ such that $\Gamma(G, S) \setminus B_N$ contains at least two different infinite connected components C_1 and C_2.

Let $(g_i)_{i\in\mathbb{N}} \subset C_1$ and $(h_i)_{i\in\mathbb{N}} \subset C_2$ be sequences with no repeated elements. Let $Y \subset \{0,1,2\}^G$ defined as $Y = Y_1 \cap Y_2$ where:

$$Y_1 = \{y \in \{0,1,2\}^G \mid 2 \in \{y_g, y_h\} \implies g = h\}$$
$$Y_2 = \{y \in \{0,1,2\}^G \mid y_g = 2 \implies \forall n \in \mathbb{N}, y_{gg_n} = y_{gh_n}\}$$

Analogously to the proof of Theorem 2.16, if the sequences are recursive with oracle $\texttt{WP}(G)$ then Y is effectively closed. We claim such sequences exist.

Fix a total order on S and extend it to a lexicographic order in S^*. Let N be as above and let $w_0 \in S^*$ such that $w_0 =_G g_0 \in C_1$. Consider the Turing machines T_g with oracle $\texttt{WP}(G)$ that on input $n \in \mathbb{N}$:

- If $n = 0$ returns w_0.
- Let $M = N + n + |w_0|$. Solve the word problem for every $w \in S^*$ such that $|w| \leq 2M$. This allows to construct B_M of $\Gamma(G, S)$.
- Let H_{g_0} be the connected component of $B_M \setminus B_N$ which contains g_0.
- Assign the value 0 to every element of $H_{g_0} \setminus \{w_0\}$ and 1 to w_0. Assign $g_1, \dots, g_n$ to ϵ. And initiate a variable k with its value set initially to 1.
- While $k \leq n$ do the following: Iterate over all $w \in S^*$ lexicographically. If $w_0 w$ has the value 0 and belongs to H_{g_0} then:
  - Turn the value $w_0 w$ to 1.
  - Assign $g_k = w_0 w$ and increase k by 1.
- Return g_n.

As the component C_1 is infinite, the value of M suffices to find n different elements. It is clear this machine yields a sequence of distinct elements in component C_1. The machine T_h for the sequence in the component C_2 is analogous.

Suppose Y is sofic. As in Theorem 2.16 we can consider an SFT extension $X \subset \mathcal{B}^G$ given by a 1-block code $\phi : X \twoheadrightarrow Y$. Let also $M \in \mathbb{N}$ be a bound such that the union of all the supports of one finite set of forbidden patterns defining X is contained in B_M. Let $L = N + M$.

As G is finitely generated $|B_L| < \infty$. Consider thus the finite set $\mathcal{P} = \{p \in \mathcal{B}^{B_L} \mid \phi([p]_{1_G}) \cap [2]_{1_G} \neq \emptyset\}$. Clearly $|\mathcal{P}| \leq |\mathcal{B}|^{|B_L|} < \infty$. Consider $w \in \{0,1\}^{\mathbb{N}}$ and fix $y^w \in \bigcap_{n\in\mathbb{N}} [w_n]_{g_n} \cap [2]_{1_G}$. Clearly $y^w \in Y$. As there is an infinite number of such y^w there exist $w_1 \neq w_2$ and $x^{w_1}, x^{w_2} \in X$ such that $\phi(x^{w_1}) = y^{w_1}$ and $\phi(x^{w_2}) = y^{w_2}$ and $x^{w_1}|_{B_L} = x^{w_2}|_{B_L}$.

By definition of L we have that $\widetilde{x} \in X$ where:

$$\widetilde{x}_g = \begin{cases} (x^{w_1})_g, \text{ if } g \in C_1 \\ (x^{w_2})_g, \text{ if } g \in G \setminus C_1 \end{cases}$$

Thus $\widetilde{y} = \phi(\widetilde{x})$ satisfies that $\widetilde{y}_{1_g} = 2$, $\widetilde{y}|_{C_1} = (y^{w_1})|_{C_1}$ and $\widetilde{y}|_{C_2} = (y^{w_2})|_{C_2}$. Let $n \in \mathbb{N}$ such that $(w_1)_n \neq (w_2)_n$ Then: $\widetilde{y}_{g_n} = (y^{w_1})_{g_n}$ and $\widetilde{y}_{h_n} = (y^{w_2})_{h_n} = (y^{w_2})_{g_n}$. Therefore $\widetilde{y} \notin Y_2$ which implies that $\widetilde{y} \notin Y$. □

3. *G*-machines

Classical Turing machines keep their information in a bi-infinite tape, and are only able to work on inputs which are codified in the form of words. While in $\mathbb{Z}$ this is a natural model to study subshifts, it becomes cumbersome in general groups as we are forced to introduce pattern codings. Moreover, as we saw in Section 2, there is a number of constraints to what can be done with Turing machines when $\texttt{WP}(G)$ is undecidable, and a general setting forces the use of oracles.

In this section we introduce an alternative model of computation which we call a G-machine. In this model, the tape is replaced by a finitely generated group G. These machines receive patterns $p \in \mathcal{A}^*_G$ as input instead of words and move by using the set S of generators. Similar machines using Cayley graphs as a tape have already been mentioned in [12] and studied in more detail in [8], but these machines take their input as a word in an auxiliary tape and only use the graph as a working tape. Another work considering subshifts defined by one or more Turing machine heads walking on the group has also been done in [20].

We begin this section by defining G-machines and the classes of languages they define. Then we present some robustness results similar to the ones satisfied by classical Turing machines. As the main result of this section, we characterize the class of G-effectively closed subshifts as those whose set of forbidden patterns is G-recursively enumerable, hence giving a characterization of this class without the use of oracles. We end this section with two applications of these machines: one to the domino problem of general groups (Theorem 3.7) and another in the form of a simulation theorem (Theorem 3.8).

Definition. A *G-machine* is a 6-tuple $(Q, \Sigma, \sqcup, q_0, Q_F, \delta)$ where Q is a finite set of states, Σ is a finite alphabet, $\sqcup \in \Sigma$ is the blank symbol, $q_0 \in Q$ is the initial state, $Q_F \subset Q$ is the set of accepting states and $\delta : \Sigma \times Q \to \Sigma \times Q \times S$ is the transition function.

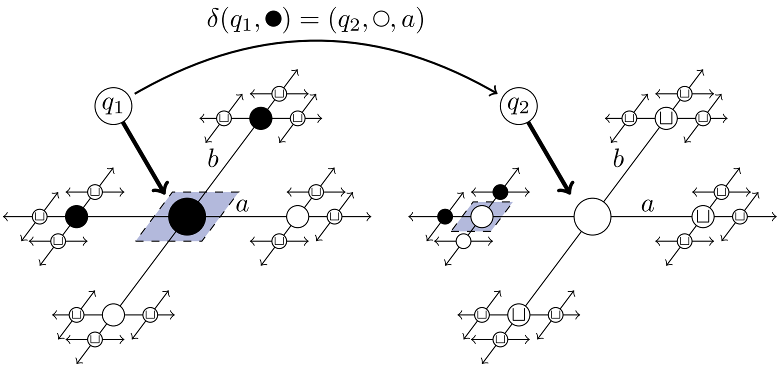


Fig. 2. A fixed head transition of an F_2-machine.

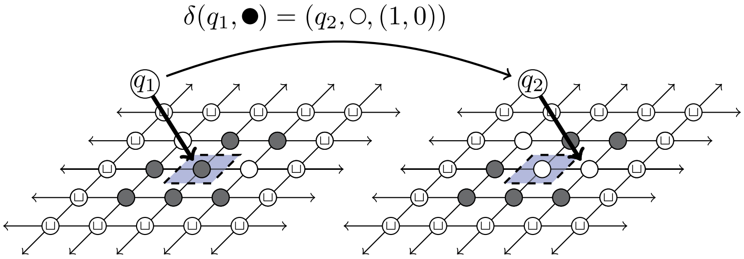


Fig. 3. A moving head transition of a $\mathbb{Z}^2$-machine.

As in the case of Turing machines, we can define the action of a Turing machine in two different ways. We call these the fixed head and moving head models.

In the fixed head model, a G-machine T acts on the set $\Sigma^G \times Q$ as follows: let $(x,q) \in \Sigma^G \times Q$ and $\delta(x_{1_G}, q) = (a, q', s)$. Then $T(x,q) = (\sigma_{s^{-1}}(\widetilde{x}), q')$ where $\widetilde{x}|_{1_G} = a$ and $\widetilde{x}|_{G\setminus\{1_G\}} = x|_{G\setminus\{1_G\}}$. Fig. 2 illustrates this action when G is a free group. Here the head of the Turing machine is assumed to stay at a fixed position and the tape moves instead.

In the moving tape model, a G-machine T acts on the set $\Sigma^G \times G \times Q$ as follows: let $(x,g,q) \in \Sigma^G \times G \times Q$ and $\delta(x_{1_G}, q) = (a, q', s)$. Then $T(x,g,q) = (\widetilde{x}, gs, q')$ where $\widetilde{x}|_{1_G} = a$ and $\widetilde{x}|_{G\setminus\{1_G\}} = x|_{G\setminus\{1_G\}}$. Fig. 3 illustrates this action when G is $\mathbb{Z}^2$. Here the tape remains fixed and the second coordinate keeps track of the position of the head.

Let $F \subset G$ be a finite set and $p \in \Sigma^F$. Let $x^p \in \Sigma^G$ be the configuration such that $(x^p)|_F = p$ and $(x^p)|_{G\setminus F} \equiv \sqcup$. We say that T *accepts* p if there is $n \in \mathbb{N}$ such that $T^n(x^p, q_0) \in \Sigma^G \times Q_F$ in the fixed head model or equivalently $T^n(x^p, 1_G, q_0) \in \Sigma^G \times G \times Q_F$ in the moving head model. $L \subset \Sigma^*_G$ is *G-recursively enumerable* if there exists a G-machine T which accepts $p \in \Sigma^*_G$ if and only if $p \in L$. If both L and $\Sigma^*_G \setminus L$ are G-recursively enumerable we say L is *G-decidable.*

So far we have defined these machines using a fixed set of generators S. In the next proposition we show that the languages defined by such machines do not depend on this arbitrary choice.

Proposition 3.1. *Let S, S' be finite subsets of G such that $\langle S \rangle = \langle S' \rangle = G$. Let $L \subset \mathcal{A}^*_G$ be recursively enumerable using S' as the movement set. Then L is recursively enumerable using S.*

Proof. Let $T_{S'}$ be a G-machine using S' as the movement set recognizing L. As $\langle S \rangle = G$ each $s' \in S'$ can be written as $s' = s_1 \dots s_{n(s')}$ where every $s_i \in S$. Consider T_S as a copy of $T_{S'}$ where for each state $q \in Q$ we add a copy q_{s',s_i} for $s' \in S$ and $i \in \{1, \dots, n(s')\}$, and every instruction $\delta(a,q) = (b,r,s')$ in $T_{S'}$ is replaced with the instructions:

- $\delta(a,q) = (b, r_{s',s_1}, s_1)$
- $\forall a \in \Sigma$ and $1 \leq i < n(s)$, $\delta(a, r_{s',s_i}) = (a, r_{s',s_{i+1}}, s_{i+1})$
- $\forall a \in \Sigma$, $\delta(a, r_{s',s_{n(s')}}) = (a, r, 1_G)$.

The modified machine T_S moves with the set of generators S and accepts the same patterns as $T_{S'}$. □

The class of G-machines shares also the robustness of Turing machines with respect to slight changes in its definition. For example, we can allow multiple tapes with multiple independent writing heads. We shall briefly and informally define this model as it will be used as a tool in a proof later on.

A *multiple head G-machine* is the same as a G-machine, except that the machine uses G^n as a tape and the transition function is $\delta : \Sigma^n \times Q \to \Sigma^n \times Q \times S^n$, where n is the number of heads of the machine. The action of this machine is defined analogously as before in either the moving head or moving tape model. It *accepts* a pattern $p \in \mathcal{A}_G^*$ if starting from the initial configuration $((x^p, \sqcup^G, \dots, \sqcup^G), q_0)$ the machine reaches in a finite number of steps an accepting state in Q_F.

In these machines each head works on its own tape, but can "read" the content of other tapes. By codifying independent movements of a tape accordingly, it is able to read not only what each head is looking at a certain step but what is written in an arbitrary finite portion of the other tapes.

Proposition 3.2. *Let $L \subset \Sigma_G^*$. There exists a multiple head G-machine which accepts exactly patterns $p \in L$ if and only if L is G-recursively enumerable.*

This extended model is useful to prove the second of the following two results which links oracle machines to G-machines. The first result is relatively straightforward, as G-machines can be simulated by a machine with oracle $\mathtt{WP}(G)$ by creating arbitrarily big balls of the Cayley graph. The second result is more interesting as it says that oracle machines can be simulated by G-machines.

Theorem 3.3. *Let $L \subset \Sigma_G^*$ be G-recursively enumerable. Then there exists a recursively enumerable with oracle $\mathtt{WP}(G)$ set of pattern codings $\mathcal{C}$ such that $L = \mathsf{p}(\mathcal{C})$.*

Proof. Suppose T_G is the G-machine recognizing L. With an oracle of $\mathtt{WP}(G)$, a machine can construct balls B_n of $\Gamma(G, S)$ for arbitrary n. A codification of B_n allows a classical Turing machine to simulate at least n applications of T_G in the moving head model as the head starts in the origin and moves at most one generator per iteration. Let T be the Turing machine with oracle $\mathtt{WP}(G)$ which does the following on input c.

- Let $N = 2\max_{(w,a)\in c} |w|$. Solve the word problem for all $w \in S^*$ of length at most N. If c is inconsistent accept.
- Let $k = N$ and iterate the following procedure: Solve the word problem for $w \in S^*$ of length at most k and simulate T_G over $\mathsf{p}(c)$ for k steps. If this procedure accepts then accept, otherwise increase k by 1.

Clearly, T accepts c if and only if either c is inconsistent or $\mathsf{p}(c) \in L$. □

Definition. A language $L \subset \Sigma_G^*$ is said to be *closed by extensions* if for each $p_1 \in \Sigma^{F_1}$, $p_2 \in \Sigma^{F_2}$ such that $F_1 \subset F_2$ and $p_2|_{F_1} = p_1$ then $p_1 \in L \implies p_2 \in L$.

Theorem 3.4. *Let G be an infinite group and $\mathcal{C}$ a recursively enumerable with oracle $\mathtt{WP}(G)$ set of pattern codings. If $\mathsf{p}(\mathcal{C})$ is closed by extensions, then $\mathsf{p}(\mathcal{C})$ is G-recursively enumerable.*

Proof. Without loss of generality we can suppose $\mathcal{C}$ is a maximal set of pattern codings which gives $\mathsf{p}(\mathcal{C})$. Moreover we can also assume that T is a one-sided Turing machine with a reading tape and a working tape.

The construction is a multiple head G-machine $\mathcal{M}$ which consists of the following six layers (see Fig. 4):

1. A storage layer $\mathcal{M}_{\mathrm{STORE}}$ where the input $p \in \Sigma_G^*$ is stored.
2. A machine $\mathcal{M}_{\mathrm{PATH}}$ which constructs an arbitrarily long one-sided non-intersecting path starting from 1_G.
3. A machine $\mathcal{M}_{\mathrm{VISIT}}$ which is able to visit iteratively all the elements of B_n for $n \in \mathbb{N}$ starting with n initially assigned to 1.
4. A Machine $\mathcal{M}_{\mathrm{ORACLE}}$ which solves $\mathtt{WP}(G)$.
5. An auxiliary layer $\mathcal{M}_{\mathrm{AUX}}$ which serves as a nexus between the first layer and the sixth.
6. A simulation layer $\mathcal{M}_{\mathrm{SIM}}$ which simulates T in the one-sided path created by $\mathcal{M}_{\mathrm{PATH}}$.

We will first describe $\mathcal{M}_{\mathrm{PATH}}$ and $\mathcal{M}_{\mathrm{VISIT}}$ which are the most complicated components. Then we will describe the general workings of the machine.

We begin by describing $\mathcal{M}_{\mathrm{PATH}}$ in detail. Let the set of generators $S = \{g_1, g_2, \dots, g_k\}$ and consider the G-machine $\mathcal{M}_{\mathrm{PATH}} := (Q, \Sigma, \sqcup, q_0, Q_F \delta)$ where $Q := \{I, B\} \cup (S \times \{\leftarrow, \rightarrow\})$, $\Sigma = (\{\sqcup, \rhd\} \cup S) \times \{\sqcup, \otimes\} \times (\{\sqcup\} \cup S)$, $q_0 = I$, $Q_F = \emptyset$ (we force the machine to loop), and δ is given by the following rules where $*_i$ stands for an arbitrary fixed symbol.

$$\delta((\sqcup, \sqcup, \sqcup), I) = ((\rhd, \otimes, g_1), g_1^{\leftarrow}, g_1).$$

$$\delta((\sqcup, \sqcup, \sqcup), g_i^{\leftarrow}) = ((g_i, \otimes, \sqcup), g_1^{\rightarrow}, 1_G).$$

$$\delta((*_1, \otimes, *_2), g_i^{\rightarrow}) = ((*_1, \otimes, g_i), g_i^{\leftarrow}, g_i).$$

$$\delta((*_1, \otimes, *_2), g_i^{\leftarrow}) = ((*_1, \otimes, *_2), B, g_i^{-1}).$$

$$\delta((g_j, \otimes, g_i), B) = \begin{cases} ((g_j, \otimes, g_i), g_{i+1}^{\rightarrow}, 1_G), & \text{if } i < k \\ ((\sqcup, \sqcup, \sqcup), B, g_j^{-1}), & \text{if } i = k. \end{cases}$$

$$\delta((\rhd, \otimes, g_i), B) = ((\rhd, \otimes, g_i), g_{i+1}^{\rightarrow}, 1_G), \text{ if } i < k$$

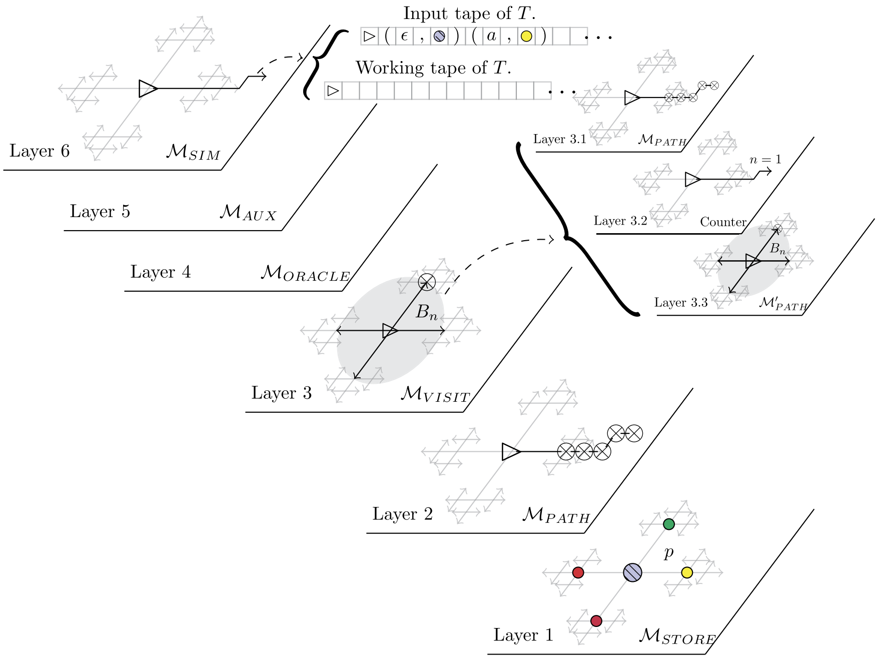

Input tape of T.
▷ (ϵ ,) (a ,) . . .
Working tape of T.
▷ . . .
Layer 6 $\mathcal{M}_{SIM}$
Layer 5 $\mathcal{M}_{AUX}$
Layer 4 $\mathcal{M}_{ORACLE}$
Layer 3 $\mathcal{M}_{VISIT}$
B_n
Layer 3.1 $\mathcal{M}_{PATH}$
$n = 1$
Layer 3.2 Counter
B_n
Layer 3.3 $\mathcal{M}'_{PATH}$
Layer 2 $\mathcal{M}_{PATH}$
p
Layer 1 $\mathcal{M}_{STORE}$

Fig. 4. Construction of the machine $\mathcal{M}$ as a multiple head G-machine.

The rules from δ codify a backtracking in G which marks a one-sided non-intersecting infinite path in G. The states I and B stand for initialization and backtracking respectively. The elements from Σ are triples (a_1, a_2, a_3) which indicate the following information: my left and right neighbors are a_1 and a_3 respectively and I belong to the path if $a_2 = \otimes$. The first rule initializes the infinite path by using the symbol $\rhd$ to indicate that there is no element to the left, marks the identity of the group as part of the path by using $\otimes$ and sets the next element in the direction g_1. The second and third rules mark the left and right neighbors respectively and move to the next position. Rule 4 deals with the case of reaching a position already marked and going back. Rule 5 and 6 search the next available direction which potentially admits an infinite path and backtrack if every position has already been searched. Rule 6 lacks a case where $i = 2k$ on purpose because such a state is never reached as the group is infinite.

Next we describe $\mathcal{M}_{\text{VISIT}}$ that visits all elements of every ball B_n in G iteratively. It suffices to construct it as a multiple head G-machine with three layers as follows. The first layer runs a copy of $\mathcal{M}_{\text{PATH}}$. The second layer makes use of the path defined by $\mathcal{M}_{\text{PATH}}$ to simulate a counter which has value $n \in \mathbb{N}$ – any one-sided Turing machine can be simulated in the path by identifying the instructions L, R with the first and third coordinates of Σ. The third layer runs another copy of $\mathcal{M}_{\text{PATH}}$, which is allowed only to run over words of length n. This is achieved by using the counter in second layer to measure the length of the path visited by the third layer and restrict it to be less than n. Each time the whole ball B_n is visited (that is, $((\rhd, \otimes, g_k), B)$ is reached in the third layer) then the counter in the second layer increments n by 1 and the third layer starts anew.

If at a given time the first layer, which constructs the one-sided path, backtracks until reaching a cell used by the counter in the second layer, then the second and third layers are erased and restart. As the group is infinite, then by choosing an adequate number of computation steps, the path generated by $\mathcal{M}_{\text{PATH}}$ in the first layer is arbitrarily long. Thus the head of the third tape is able to visit every element of B_n for arbitrarily big n.

Finally, we describe the overall functioning of $\mathcal{M}$:

- The input $p \in \Sigma_G^*$ is stored in $\mathcal{M}_{\text{STORE}}$ whose head mimics that of $\mathcal{M}_{\text{VISIT}}$ without changing anything.
- The machines $\mathcal{M}_{\text{PATH}}$ and $\mathcal{M}_{\text{VISIT}}$ run independently.
- $\mathcal{M}_{\text{SIM}}$ uses the path given by $\mathcal{M}_{\text{PATH}}$ to simulate two one-sided Turing machine tapes: an input tape where input will be stored, and a working tape which simulates T over that input.
- If at any moment the working tape of $\mathcal{M}_{\text{SIM}}$ makes a call to the oracle $\texttt{WP}(G)$, then $\mathcal{M}_{\text{ORACLE}}$ is made to mark the origin, follow the path $w \in S^*$ and accept the call if the last symbol is marked. Then it erases everything and goes back to the origin.
- Whenever $\mathcal{M}_{\text{VISIT}}$ arrives at a position where the first layer is not marked by $\sqcup$, the head at $\mathcal{M}_{\text{AUX}}$ follows the path w marked from 1_G by the first layer of $\mathcal{M}_{\text{VISIT}}$ and writes (w, a) in the input tape of $\mathcal{M}_{\text{SIM}}$. Then $\mathcal{M}_{\text{AUX}}$ marks position w as already visited and returns to 1_G.

- If at a given time $\mathcal{M}_{\text{AUX}}$ extends the pattern coding written in the reading tape of the fifth layer, then the working tape of $\mathcal{M}_{\text{SIM}}$ erases everything and begins anew.
- If at any moment the end of the simulated path created by $\mathcal{M}_{\text{PATH}}$ backtracks into a cell used by the written portion of $\mathcal{M}_{\text{SIM}}$, then the content of all tapes except $\mathcal{M}_{\text{PATH}}$ and $\mathcal{M}_{\text{STORE}}$ is erased and they start anew.
- $\mathcal{M}$ accepts if and only if the working tape of $\mathcal{M}_{\text{SIM}}$ does.

As $\mathcal{M}_{\text{PATH}}$ is able to construct arbitrarily long one-sided and non-intersecting paths, there is a finite number of computation steps such that $\mathcal{M}_{\text{VISIT}}$ will visit all of the support of p. Thus the fourth layer will write a consistent pattern coding c such that $p = \mathtt{p}(c)$ which is accepted by the working tape of $\mathcal{M}_{\text{SIM}}$ if and only if $p \in \mathtt{p}(\mathcal{C})$ (as $\mathcal{C}$ is maximal). By considering a path which has length at least two times the running time of all the other algorithms, this eventually happens. Conversely, if $p \notin \mathtt{p}(\mathcal{C})$, as $\mathtt{p}(\mathcal{C})$ is closed by extensions, the acceptance of any partial coding c' would mean that $p \in \mathtt{p}(\mathcal{C})$, therefore, the machine never accepts. □

Corollary 3.5. *A subshift $X \subset \mathcal{A}^G$ is G-effectively closed if and only if there exists a G-recursively enumerable set $\mathcal{F} \subset \mathcal{A}^*_G$ such that $X = X_{\mathcal{F}}$.*

Proof. As X is G-effectively closed, the set of forbidden pattern codings $\mathcal{C}$ can be chosen to be maximal. This in turn gives a maximal set of forbidden patterns $\mathtt{p}(\mathcal{C})$ which is closed by extensions. Theorems 3.3 and 3.4 imply the result. □

For a language $L \subset A^*$ we define the Turing jump L' as the halting problem for Turing machines with oracle L.
Let $\mathtt{HALT}_G = \{\langle T \rangle \mid T$ is a G-machine which accepts the empty input$\}$.

Corollary 3.6. *Let G be an infinite group. $\mathtt{HALT}_G$ is $\mathtt{WP}(G)'$-hard, that is, it is at least as hard as the halting problem for Turing machines with oracle $\mathtt{WP}(G)$.*

Proof. Let T be a Turing machine with oracle $\mathtt{WP}(G)$. Consider the construction from Theorem 3.4 without the Visit and Auxiliary tapes. Thus, there is only the tape which searches the infinite path, the oracle layer, and the layer which simulates T (now only on empty input). It is clear that this machine accepts the empty input (and all inputs) if and only if T accepts the empty input. □

Corollary 3.5 implies that G-effectively closed subshifts can be defined either by oracle machines or by G-machines. This nice characterization allows us to simulate Turing machines in groups which may not even have torsion-free elements. In what remains of this section we present applications of these machines both to the domino problem and to construct a simulation theorem.

3.1. Application: a class of groups with undecidable domino problem

The *domino problem* of a finitely generated group G is defined as the language given by the finite sets of pattern codings which give an empty subshift. Informally:

$$\mathtt{DP}(G) = \{\langle \mathcal{F} \rangle \mid |\mathcal{F}| < \infty, X_{\mathcal{F}} = \emptyset\},$$

where $\langle \mathcal{F} \rangle$ is a codification of the finite set of forbidden patterns $\mathcal{F}$. Another related notion is the *origin constrained domino problem*, where a symbol in the alphabet is fixed to appear at the origin.

$$\mathtt{OCDP}(G) = \{(a, \langle \mathcal{F} \rangle) \mid |\mathcal{F}| < \infty, X_{\mathcal{F}} \cap [a]_{1_G} = \emptyset\}$$

It is known that both problems are decidable in $\mathbb{Z}$ [17] but undecidable in $\mathbb{Z}^d$ with $d > 1$ [5,22]. Clearly, the decidability of $\mathtt{OCDP}(G)$ implies the decidability of $\mathtt{DP}(G)$ as it would suffice to run the algorithm for every symbol of the finite alphabet. So far, we do not know any group where the decidability of these two languages differ. In this section we use G-machines to exhibit a class of groups where these problems are undecidable.

Theorem 3.7. *Let G be an infinite group with the special symbol property. Then:*

1. *The origin constrained domino problem $\mathtt{OCDP}(G \times \mathbb{Z})$ is $\mathtt{WP}(G)'$-hard.*
2. *For any non-trivial finite group H, the domino problem for $(G \times \mathbb{Z}) * H$ is $\mathtt{WP}(G)'$-hard.*

Proof. Let $G = \langle S \rangle$ and T be a G-machine with tape alphabet $\Sigma = \{\sqcup, 0, 1\}$ and transition function $\delta : \Sigma \times Q \to \Sigma \times Q \times S$. Denote the states by $Q = \{1, \dots, k\}$ where the initial and final states are 1 and k respectively. Finally, let $\mathcal{A} = \Sigma \times \{0, \dots, k\}$ and $Z = \Sigma^G \times X_{\leq k} \subset \mathcal{A}^G$ where:

$$X_{\leq k} = \{x \in \{0, \dots, k\}^G \mid 0 \notin \{x_g, x_h\} \implies g = h\}.$$

The subshift Z consists on configurations where there is at most one appearance of a state in Q. As G satisfies the special symbol property, this is a sofic subshift.

We are going to define an extended sofic subshift $Y \subset \mathcal{A}^{G\times\mathbb{Z}}$ which simulates the dynamical behavior of T by letting G act as the space and $\mathbb{Z}$ as the time direction. We do this by defining its set of forbidden patterns as $\mathcal{F} = A_1 \cup A_2 \cup A_3 \cup A_4$ where these four sets are defined as follows:

- Let $F \subset G$. For $p \in \mathcal{A}^F$ we define its immersion $\gamma(p) : \mathcal{A}^F \to \mathcal{A}^{F\times\{0\}}$ where $\gamma(p)_{(g,0)} = p_g$ for every $g \in F$. We define A_1 as the immersion of the forbidden patterns defining Z.
- Consider the support $F = \{(1_G, 0), (1_G, 1)\}$. We define A_2 as the set of $p \in \mathcal{A}^F$ such that $p_{(1_G,0)} = (a, 0)$ and $p_{(1_G,1)} = (b, \cdot)$ with $b \neq a$.
- Let $\delta(a, q) = (b, r, s)$. We define $A_3 = B_1 \cup B_2$ where these sets are the following:
  - Let $F = \{(1_G, 0), (1_G, 1)\}$, we define B_1 as the set of $p \in \mathcal{A}^F$ such that $p_{(1_G,0)} = (a, q)$ and $p_{(1_G,1)} = (c, \cdot)$ with $c \neq b$.
  - Let $F_s = \{(1_G, 0), (s, 1)\}$, we define B_2 as the set of $p \in \mathcal{A}^{F_s}$ such that $p_{(1_G,0)} = (a, q)$ and $p_{(s,1)} = (\cdot, t)$ with $t \neq r$.
- We define A_4 to be the patterns with support $\{(1_G, 0)\}$ containing the symbol (a, k) for some $a \in \Sigma$.

This subshift is clearly sofic, as its forbidden patterns are the immersion of the forbidden patterns of a sofic subshift plus a finite amount of new forbidden patterns. The set A_1 just forces every coset (G, z) to contain a configuration of Z. Said otherwise, at most one head. A_2 forces that whenever a state 0 appears then the symbol must remain unchanged. A_3 is composed of two rules related to the head: the first, B_1 forces the symbol in the head position to correspond to the one from the rule δ. B_2 forces the movement of the head to correspond to the rule δ. Finally, A_4 forbids the appearance of the final state k.

Consider the coding $\rho : \Sigma^G \times G \times Q \to Z$ given by $\rho(x, h, q) = (x, z)$, where $z_h = q$ and $z|_{G\setminus\{h\}} \equiv 0$. This coding takes a configuration in the moving tape model and represents it as an element of the subshift Z. The rules defining Y force that if $y \in Y$ and $y_{(g,n)} = \rho(x, h, q)_g$ for all $g \in G$, for all $m \geq 0$ one has $y_{(g,n+m)} = \rho(T^m(x, h, q))_g$.

Using this previous relation and the fact that appearances of the final state are forbidden, we obtain that there exists $y \in Y$ such that $\forall g \in G$ then $y_{(g,0)} = \rho(\sqcup^G, h, 1)$ for some $h \in G$ if and only if T does not accept the empty input.

Let $X_{\text{aux}} \subset \{0, \star\}^{G\times\mathbb{Z}}$ be the SFT defined by the following forbidden patterns: for every $s \in S$ the pattern $p \in \{0, \star\}^{\{1_H, s\}}$ such that $p_{1_H} = \star$ but $p_s \neq \star$ is forbidden. This basically means that if a $\star$ appears in a position, then the whole G-coset contains a $\star$.

Now we have all the elements for the final construction: let $X_{\text{final}} \subset \mathcal{A}^{G\times\mathbb{Z}} \times X_{\text{aux}}$ defined by the following forbidden patterns:

- The immersion of all forbidden patterns of Y.
- A $\star$ in X_{aux} must always be accompanied by $(\sqcup, j)$ for some $j \in \{0, \dots, k\}$.

This subshift is again sofic, since its forbidden patterns are the immersion of those of Y and a finite number of forbidden symbols in the alphabet. The role of X_{aux} is to force a G-coset to represent the machine T starting on empty input. We claim that $X_{\text{final}} \cap [((\sqcup, 1), \star)]_{1_G} = \emptyset$ if and only if T accepts the empty input.

Indeed, let $x \in X_{\text{final}} \cap [((\sqcup, 1), \star)]_{1_G}$. By using the definition of X_{aux}, the local rule of X_{final} and the characterization of Y, we deduce that for all $g \in G \setminus \{1_G\}$ then $x_{(g,0)} = ((\sqcup, 0), \star)$. Therefore the projection π_1 to the first coordinate of x would satisfy $\pi_1(x)|_{(g,0)} = \rho(\sqcup^G, h, 1)_g$. This implies that $\pi_1(x)|_{h(g,k)} = \rho(T^m(\sqcup^G, h, 1))_g$ for all $m \geq 0$. As the final state k cannot appear, we conclude that T does not accept the empty input. Conversely, if T does not accept the empty input we can construct a valid point as follows: let $y \in Y$ such that $\forall n \geq 0$ $y_{(g,n)} = \rho(T^n(\sqcup^G, 1_G, 1))$ and $\forall m \leq -1$ $y_{(g,m)} = (\sqcup, 0)$. This is a valid point of Y as T does not accept the empty input. We can therefore define $x \in X_{\text{final}}$ as follows:

$$x_{(g,k)} = \begin{cases} (y_{(g,0)}, \star), & \text{if } k = 0 \\ (y_{(g,k)}, 0), & \text{if } k \neq 0 \end{cases}$$

which satisfies $x \in X_{\text{final}} \cap [((\sqcup, 1), \star)]_{1_G}$.

Note that the previous argument implies that if $((\sqcup, 1), \star)$ appears in a configuration, it can only do so in at most one position. Therefore, we can consider a 1-block SFT extension of X_{final} with at most one preimage a of $((\sqcup, 1), \star)$. This is a $G \times \mathbb{Z}$ SFT such that $X \cap [a] = \emptyset$ if and only if T accepts the empty input. Therefore $\texttt{OCDP}(G \times \mathbb{Z})$ is at least as hard as the halting problem for G-machines which in turn is $\texttt{WP}(G)'$-hard by Corollary 3.6.

Let H be a finite group and consider the subshift $Y_{\text{aux}} \subset \{0, \spadesuit\}^{(G\times\mathbb{Z})*H}$ defined by the following forbidden patterns: $p \in \{0, \spadesuit\}^H$ such that $|\{h \in H \mid p_h = \spadesuit\}| \neq 1$. This means that every coset of H must contain exactly one appearance of $\spadesuit$. In the following, we choose a configuration $y \in Y_{\text{aux}}$ every $(G \times \mathbb{Z})$-coset contains at most one occurrence of $\spadesuit$. Let $\bar{h} \in H \setminus \{1_H\}$ and w be a reduced word representation of an element in $(G \times \mathbb{Z}) * H$. Define

$$y_w = \begin{cases} 1, & \text{if } w \text{ ends by } \bar{h} \\ 0, & \text{otherwise.} \end{cases}$$

By using Y_{aux} as an extra SFT layer, we can force the appearance of $((\sqcup, 1), \star)$ every time a $\spadesuit$ appears, and immerse the patterns of X_{final} into $(G \times \mathbb{Z}) * H$. By definition, each configuration in Y_{aux} has at least one coordinate marked by an $\spadesuit$,

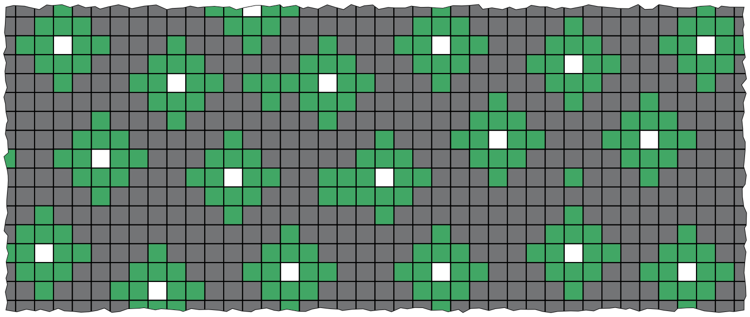


Fig. 5. Example of a configuration of Y_2 for the group $\mathbb{Z}^2$ with the canonical generators. The symbols 0, 1 and 2 are represented by the colors ■, □ and 🟩 respectively. (For interpretation of the colors in this figure, the reader is referred to the web version of this article.)

and y has also the property that each $(G\times\mathbb{Z})$-coset contains at most one occurrence of ♠. We can thus repeat the previous argument to conclude that $\texttt{DP}((G\times\mathbb{Z})*H)$ is $\texttt{WP}(G)'$-hard. □

The role of the free product with H is to ensure that the machine starts the calculation over an empty tape at some place in every configuration. In the classical construction of Robinson [22] in the plane, this property is obtained using a hierarchical construction. We do not know if a generalization of this construction can be done in general groups.

Notice also that this result does not give new groups with undecidable domino problem when G has at least one non-torsion element. Indeed, if $\mathbb{Z}$ embeds into G then $\mathbb{Z}^2$ also embeds into $G\times\mathbb{Z}$. The advantage of this method using G-machines is that it allows to give a result over torsion groups such as the Grigorchuk group [13].

3.2. Application: a simulation theorem with oracles

In [3,11] it is shown that every effectively closed subshift on $\mathbb{Z}$ can be obtained as the projective subdynamics of a sofic subshift on $\mathbb{Z}^2$. As Propositions 2.9 and 2.13 show, an analogue can not hold for arbitrary G-effectively closed subshifts when G is recursively presented as the projective subdynamics would necessarily be effectively closed. Nevertheless, using G-machines, we can obtain a similar result if we allow the addition of a particular subshift as a universal oracle to our construction. Formally we show:

Theorem 3.8. *For every finitely generated group G, there exists a $G\times\mathbb{Z}$-effectively closed subshift $U\subset\widetilde{\mathcal{B}}^{G\times\mathbb{Z}}$ such that for every G-effectively closed subshift $X\subset\mathcal{A}^G$ which contains a uniform configuration ($\exists\bar{a}\in\mathcal{A}$ such that $\bar{a}^G\in X$), there exists an alphabet $\mathcal{B}$, a finite set of forbidden patterns $\mathcal{F}$ on alphabet $\widetilde{\mathcal{B}}\times\mathcal{B}$ and a 1-block code ϕ such that:*

$$\pi_G\left(\phi\left(\left(U\times\mathcal{B}^{G\times\mathbb{Z}}\right)\setminus\bigcup_{p\in\mathcal{F},h\in G\times\mathbb{Z}}[p]_h\right)\right)=X.$$

In order to define U we need to introduce some technical constructions. Let (X,d) be a metric space and $D\subset X$. The *packing radius* of D is $r_D=\frac{1}{2}\inf\{d(x,y)\mid x,y\in D,x\neq y\}$ and the *covering radius* of D is given by $c_D=\sup\{d(x,D)\mid x\in X\}$. Notice that for each pair of different $x,y\in D$, we have $B(x,r_D)\cap B(y,r_D)=\emptyset$ and $\bigcup_{x\in D}B(x,c_D)=X$. A set with non-zero packing radius and finite covering radius is said to be Delone. Notice that by definition a Delone subset of a non-empty set must be non-empty.

We define $Y_n\subset\{0,1,2\}^G$ as the subshift defined by the following set of forbidden patterns $\mathcal{F}_n$:

- All $p\in\{0,2\}^{B(1_G,4n)}$.
- $p\in\{0,1,2\}^{B(1_G,n)}$ such that $p_{1_G}=1$ and there exists $g\in B(1_G,n)\setminus\{1_G\}$ with $p_g\neq 2$.
- $p\in\{1,2\}^F$ where F is a connected component of $\Gamma(G,S)$ and there exist $g_1,g_2\in F,g_1\neq g_2$ such that $p_{g_1}=p_{g_2}=1$.

That is, Y_n is the set of configurations y where, if we denote the set of positions marked in y by a 1 by D_y, then D_y forms a Delone set with $r_{D_y}\geq n$ and $c_{D_y}\leq 4n$. Also, each 1 is surrounded by a ball of size at least n marked by 2's and there is no path of 2's connecting two adjacent 1s. See Fig. 5 for an example in $\mathbb{Z}^2$.

Claim. $\forall n\geq 1$, *Y_n is a non-empty, G-effectively closed subshift.*

Proof. The set $\mathcal{F}_n$ can easily be recognized by a Turing machine with oracle $\texttt{WP}(G)$, so Y_n is G-effectively closed. For the non-empty part, we claim a Delone set D satisfying $r_D\geq 2n$ and $c_D\leq 4n$ always exists. Indeed, consider the restriction to

$B(1_G,k)$ for some $k\in\mathbb{N}$ and choose a maximal set $D_k\subset B(1_G,k)$ with $r_{D_k}\geq 2$. If $c_{D_k}>4n$ then the set $K:=\{g\in B(1_G,k)\mid d(D_k,g)>2n\}$ is not empty and D_k can be extended by an element of K, contradicting its maximality. Thus $c_{D_k}\leq 4n$. Now, consider the sequence of indicator functions of $(D_k)_{k\in\mathbb{N}}$ and choose an accumulation point. This limit is the indicator function of a Delone set D which satisfies the aforementioned property. Now, define $y\in\{0,1,2\}^G$ as:

$$y_g=\begin{cases}1 & \text{if } g\in D\\ 2 & \text{if } 0<d(g,D)\leq n\\ 0 & \text{else}\end{cases}$$

As $c_D\geq 2n$ and $n\geq 1$ it follows that there is no path consisting of 2's between a pair of 1's. It follows that $y\in Y_n$. □

Consider a G-machine T with alphabet Σ and set of states Q whose head never leaves a bounded support F. Using a pigeonhole argument, it can be shown that if it accepts, it must do so before $|Q|\cdot|F|\cdot|\Sigma|^{|F|}$ steps. Consider the function $\mathtt{time}:\mathbb{N}\to\mathbb{N}$ given by $\mathtt{time}(n)=n^{n^n+n+1}$. It is clearly a computable function which satisfies the following property: for any G-machine T, there exists $N\in\mathbb{N}$ such that for every $n\geq N$, if T accepts a pattern p without leaving the support $B(1_G,n)$ then it does so before $\mathtt{time}(n)$ steps. Indeed, we can always bound $B(1_G,n)\leq|S|^n$ and thus an upper bound for the maximum number of steps without leaving the support $B(1_G,n)$ is given by $|Q|\cdot|S|^n\cdot|\Sigma|^{|S|^n}$. Choosing $N\geq\max\{|Q|,|S|,|\Sigma|\}$ we get that $\forall n\geq N$ the number of steps is bounded by n^{n^n+n+1}.

We are going to construct a $\mathbb{Z}$-subshift $X_{\mathtt{time}}$ which encodes the function $\mathtt{time}$ and instructions for a Turing machine in a convenient way. Consider the alphabet $\mathcal{A}_X=\{\bullet,\star,\oplus,\triangleright\}\cup S$. Let $\widetilde{x}\in\mathcal{A}_X^{\mathbb{N}}$ be the infinite concatenation of $\{w_n\}_{n\in\mathbb{N}}$, where $w_0=\star$ and for $n\geq 1$ the word w_n is defined as follows. Let $u_1,\dots,u_{k(n)}$ be the lexicographic enumeration of all words in S^* of length at most $4n$. Then,

$$v_{j,n}=u_j\triangleright\bullet^{\mathtt{time}(n)}u_j^{-1},\text{ and } w_n=\oplus v_{0,n}v_{1,n},\dots,v_{k(n),n}$$

Example. Let $S=\{a,a^{-1}\}$ and suppose just for this example that the words are enumerated up to length n instead of $4n$, and that $\mathtt{time}(1)=2$ and $\mathtt{time}(2)=3$. Then the first symbols of $\widetilde{x}$ would be:

$$\begin{aligned}\widetilde{x}=\star\oplus\triangleright\bullet\bullet a\triangleright\bullet\bullet a^{-1}a^{-1}\triangleright\bullet\bullet a\oplus\triangleright\bullet\bullet\bullet a\triangleright\bullet\bullet\bullet a^{-1}a^{-1}\triangleright\bullet\bullet\bullet a\\ aa\triangleright\bullet\bullet\bullet a^{-1}a^{-1}aa^{-1}\triangleright\bullet\bullet\bullet aa^{-1}a^{-1}a\triangleright\bullet\bullet\bullet a^{-1}aa^{-1}a^{-1}\triangleright\bullet\bullet\bullet aa\cdots\end{aligned}$$

With the infinite word $\widetilde{x}$ in hand, we define $X_{\mathtt{time}}\subset\mathcal{A}_X^{\mathbb{Z}}$ as the subshift such that if $x\in X$ and $x_n=\star$, then for all $m\geq 0$ we have $x_{n+m}=\widetilde{x}_m$. Clearly the forbidden patterns of $X_{\mathtt{time}}$ can be recognized by a Turing machine.

Let $\widetilde{X}_{\mathtt{time}}\subset\mathcal{A}_X^{G\times\mathbb{Z}}$ be the periodic extension of $X_{\mathtt{time}}$. That is, for all $\widetilde{t}\in\widetilde{X}_{\mathtt{time}}$ and $g\in G$ we have $\widetilde{t}_{(g,k)}=\widetilde{t}_{(1_G,k)}$ and the configuration $x\in\mathcal{A}_X^{\mathbb{Z}}$ defined by $x_k=\widetilde{t}_{(1_G,k)}$ belongs to $X_{\mathtt{time}}$.

Finally, we define $U\subset\widetilde{X}_{\mathtt{time}}\times\{0,1,2\}^{G\times\mathbb{Z}}$ by a set of forbidden patterns. In order to describe this set, we denote by π_1 and π_2 the projections to the first and second coordinate respectively.

- Let $(k_n)_{n\geq 1}$ be the sequence of positions in $\widetilde{x}$ such that $\widetilde{x}_{k_n}=\oplus$. Recall that $\mathcal{F}_n$ denotes the set of forbidden patterns defining Y_n. We forbid all patterns p with support $F\ni(1_G,0)$ such that $\pi_1(p)_{(1_G,0)}=\star$ and for which there is $n\in\mathbb{N}$ such that the restriction of $\pi_2(p)$ to $F_n=\{(g,k_n)\mid(g,k_n)\in F\}$ contains a pattern in $\mathcal{F}_n$.
- We forbid all patterns p with support $F=\{(1_G,0),(1_G,1)\}$ such that $\pi_1(p)_{(1_G,1)}\in\{\triangleright,\bullet\}$ and $\pi_2(p)_{(1_G,1)}\neq\pi_2(p)_{(1_G,0)}$.
- For $s\in S$, we forbid all patterns with support $F_s=\{(1_G,0),(s,1)\}$ such that $\pi_1(p)_{(s,1)}=s$ and $\pi_2(p)_{(s,1)}\neq\pi_2(p)_{(1_G,0)}$.

In other words, these patterns use the information on the first coordinate to force a structure on the second one as follows: The n-th coordinate marked with $\oplus$ after a $\star$ must carry a configuration $y\in Y_n$ in the second coordinate. The symbols $\triangleright$ and $\bullet$ in the layer (G,m) just copy the configuration in the layer $(G,m-1)$. The symbols from S shift the whole configuration by $s\in S$. See Fig. 6.

Claim. *U is a non-empty, $G\times\mathbb{Z}$-effectively closed subshift.*

Proof. The first set of forbidden patterns is recursively enumerable with oracle $\mathtt{WP}(G)$ as (k_n) is computable and Y_n is G-effectively closed (the Turing machine accepting patterns of Y_n can be constructed universally for all $(Y_n)_{n\in\mathbb{N}}$ such that it receives $n\in\mathbb{N}$, $p\in\{0,1,2\}^G$ as an input and accepts if $[p]\cap Y_n=\emptyset$). The rest of the forbidden patterns is a finite set, therefore U is a $G\times\mathbb{Z}$-effectively closed subshift. It is non-empty as each Y_n is non-empty. □

Now that the description of U is done, we are ready to show Theorem 3.8.

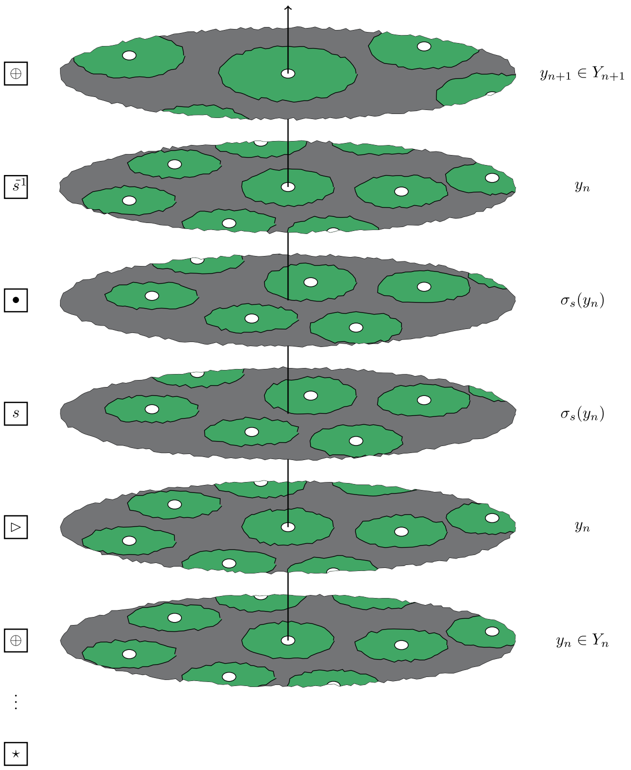


Fig. 6. A typical configuration in $U \subseteq (\{\bullet, \star, \oplus, \triangleright\} \cup S) \times \{0,1,2\}^{G\times\mathbb{Z}}$. Symbols on the left side of the picture correspond to the first coordinate of the configuration, and the part in $\{0,1,2\}^{G\times\mathbb{Z}}$ is on the right. On the example, the bottom $\oplus$ is the n-th appearance after $\star$.

Proof. Let $\mathcal{A}$ be the alphabet of X and T be the G-machine which on input $p \in \mathcal{A}_G^*$ accepts if and only if $[p] \cap X = \emptyset$. Using $\mathcal{M}_{\text{visit}}$ from Theorem 3.4 we can construct from T a machine $\widetilde{T}$ working on an infinite configuration whose description is as follows.

The machine $\widetilde{T}$ contains two tapes: a reading tape which is never modified and initially filled with symbols from $\mathcal{A}$, and a working tape. The machine $\widetilde{T}$ iterates infinitely for $n = 1, 2, \dots$ as follows: for $n \in \mathbb{N}$, the machine iterates in order $k = 1, 2, \dots, n$ the following procedure:

- Copy the pattern appearing in the reading tape in the support $B(1_G, k)$ around the head to the working tape.
- Run T over this pattern n steps. If T accepts at some point, then $\widetilde{T}$ accepts.
- Erase everything in the working tape and go back to the starting position.

Let $\Sigma \ni \sqcup$ be the alphabet of the working tape of $\widetilde{T}$ and let its set of states be $Q = \{1, \dots, k\}$, where 1 is the initial state and k the only accepting state. We proceed similarly to Theorem 3.7 by modeling this machine as a subshift on $G \times \mathbb{Z}$. We define the alphabet $\mathcal{B} = \mathcal{A} \times \Sigma \times \{0, \dots, k\}$. Here $\mathcal{A}$ is the alphabet of X, Σ is the alphabet of the working tape and $\{0, \dots, k\}$ codes the state of the head of a G-machine, 0 coding the absence of a head. In order to describe the finite set of forbidden patterns we introduce some notation. Recall that U is defined over the alphabet $\{\bullet, \star, \oplus, \triangleright\} \times \{0,1,2\}$. Therefore the set of forbidden patterns $\mathcal{F}$ is defined over the alphabet $\mathcal{A}_{\text{Final}}$ where:

$$\mathcal{A}_{\text{Final}} = \{\bullet, \star, \oplus, \triangleright\} \times \{0,1,2\} \times \mathcal{A} \times \Sigma \times \{0, \dots, k\}.$$

We denote the projection to each of these five coordinates by $\pi_1, \dots, \pi_5$ respectively. The forbidden patterns in $\mathcal{F}$ belong to four categories: *configuration patterns*, *starting patterns*, *ending patterns* and *transitions patterns*.

The *configuration patterns* force that every $\mathbb{Z}$-coset sees the same symbol in the third coordinate. Said otherwise, the third coordinate is invariant under the action of $\mathbb{Z}$. To obtain this we forbid all p with support $\{(1_G, 0), (1_G, 1)\}$ such that $\pi_3(p_{(1_G,0)}) \neq \pi_3(p_{(1_G,1)})$.

The *starting patterns* are defined by forbidding symbols in $\mathcal{A}_{\text{Final}}$ in a way such that every time the symbol $\triangleright$ appears in a G-coset, then the working tape symbols are empty (that is, marked by $\sqcup$) and all positions marked by 1 carry a head with the initial state. Formally, we force that all $a \in \mathcal{A}_{\text{Final}}$ such that $\pi_1(a) = \triangleright$ must also satisfy $\pi_4(a) = \sqcup$. Furthermore, if $\pi_2(a) = 1$ then $\pi_5(a) = 1$ and if $\pi_2(a) \in \{0, 2\}$ then $\pi_5(a) = 0$.

The *ending patterns* are described by forbidding the appearance of any symbol containing the accepting state k. Formally, every symbol $a \in \mathcal{A}_{\text{Final}}$ $\pi_5(a) = k$ is forbidden.

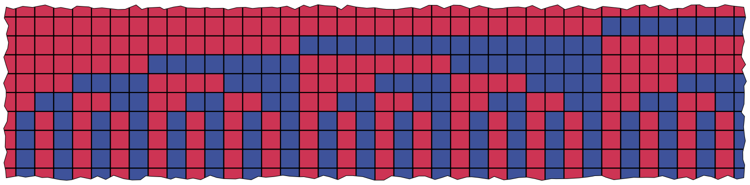


Fig. 7. A sofic subshift which doubles its period.

The *transition patterns* describe the evolution of $\widetilde{T}$ after a symbol $\triangleright$. Each time the symbol $\bullet$ appears it marks that the G-machines must execute one step with respect to the previous G-coset. The description of these patterns is the same as the one done in Theorem 3.7 with one difference. We update the tape according to the transition function of $\widetilde{T}$ only if a head is lying in a position not marked by a 0 in the second coordinate. If this happens, then the tape does not evolve.

Finally, we describe the 1-block code ϕ. Let $\bar{a} \in \mathcal{A}$ be a symbol such that $\bar{a}^G \in X$. We define a local function $\Phi : \mathcal{A}_{\text{Final}} \to \mathcal{A}$ by:

$$\Phi(a) = \begin{cases} \pi_3(a) & \text{if } \pi_1(a) = \star \\ \bar{a} & \text{otherwise} \end{cases}$$

and we set $\phi(x)_{(g,k)} = \Phi(x_{(g,k)})$.

Let $x \in \mathcal{A}^G$ be the G-projective subdynamics of $\phi(z)$, where $z \in U \times \mathcal{B}^{G\times\mathbb{Z}}$ and avoids all forbidden patterns in $\mathcal{F}$. By definition of U, as $\widetilde{X}_{\mathtt{time}}$ is a periodic extension, each G-coset of z is either completely marked by $\star$ or does not contain a $\star$ at all. If this last case happens, then $x = \bar{a}^G \in X$. Otherwise $\pi_1(z)_{(g,0)} = \star$ and thus by definition of U we have $\pi_1(z)_{(g,k)} = \widetilde{x}_k$. Suppose $x \notin X$, then there exists a ball B_n and $p \in \mathcal{A}^{B_n}$ such that $[p] \cap X = \emptyset$. This implies that T accepts the input p in a finite number of steps n_T. By definition, $\widetilde{T}$ also accepts all configurations in $[p]$ in a number of steps bounded by a function of n_T. Let B_m be a ball such that $\widetilde{T}$ never leaves B_m when working on $[p]$ (one could take for instance m as the bound on the number of steps). Let $N \geq \max\{|Q|, |S|, |\Sigma|, m\}$. Then we know that $\widetilde{T}$ starting on position 1_G would accept an input in $[p]$ in less than $\mathtt{time}(N)$ steps. Consider k_N the position of the N-th appearance of $\otimes$ in $\widetilde{x}$. By definition we know that in the G-coset in k_N, the second coordinate contains a configuration $y \in \{0,1,2\}^G$ such that $y \in Y_N$. Therefore, there exists $g \in B(1_G, 4N)$ such that $y_g = 1$. As each word of length smaller or equal to $4N$ appears, then a codification of g^{-1} eventually does. Using the rules of U, this means that after this word the next coset is marked by $\triangleright$, and the configuration in the second coordinate is $y' = \sigma_{g^{-1}(y)}$ thus $y'_{1_G} = 1$. By definition of $\widetilde{x}$, the next $\mathtt{time}(N)$ cosets are marked by $\bullet$ thus simulating $\widetilde{T}$ for that number of steps as long as the head does not see a 0 in the second coordinate. As there is a ball of size at least N around the identity marked by a symbol 2, then $\widetilde{T}$ is run for $\mathtt{time(N)}$ steps, thus reaching the accepting state k which is forbidden. This contradicts that $x \notin X$.

Conversely, each $x \in X$ can be obtained by constructing a configuration z such that $\pi_3(z)_{(g,k)} = x_g$ and $\pi_1(z)_{(g,0)} = \star$. By definition of $\widetilde{T}$ and similar arguments as above, this configuration can be completed for all $g \in G$ and $k \geq 0$ without producing forbidden patterns. For $k \leq 0$ we can just fill the coordinate (g,k) with the symbol $(\bullet, 0, x_g, \sqcup, 0)$ without creating forbidden patterns. □

We remark that the condition that X must contain a uniform configuration can easily be replaced by weaker statements. For example, it suffices to contain a periodic configuration or more generally, a G-SFT Y such that $Y \subset X$. In the proof above it would suffice to add a $\mathbb{Z}$-periodic extension of Y as an extra coordinate and change the definition of the 1-block code ϕ such that it projects to this coordinate instead of $\bar{a}$.

Another interesting aspect of this construction is that even if the subshift U is $G \times \mathbb{Z}$-effectively closed in general, it can sometimes be forced to be a sofic subshift. For example, if $G = \mathbb{Z}^d$ then $X_{\mathtt{time}}$ is an effectively closed $\mathbb{Z}$-subshift and thus its periodic extension is a sofic $\mathbb{Z}^{d+1}$-subshift by [3,11]. Also, we remark that in the second coordinate of U, it suffices to contain a non-empty subsystem of Y_n in each G-coset. For $\mathbb{Z}^d$ it is not hard to produce sofic subshifts with those properties. For example, the subshift shown in Fig. 7 in which each horizontal strip contains a periodic configuration which doubles its period when advancing vertically can be easily shown to be sofic and adapted by adding extra symbols to produce a suitable subsystem of the second layer of U.

Acknowledgements

This work was partially supported by the ANR project QuasiCool (ANR-12-JS02-011-01) and the ANR project CoCoGro (ANR-16-CE40-0005).

References

[1] Nathalie Aubrun, Mathieu Sablik, Row-constrained effective sets of colourings in the 2-fold horocyclic tessellations of $\mathbb{H}^2$ are sofic, arXiv:1602.04061.

[2] Nathalie Aubrun, Jarkko Kari, Tiling problems on Baumslag–Solitar groups, preprint, 2013.
[3] Nathalie Aubrun, Mathieu Sablik, Simulation of effective subshifts by two-dimensional subshifts of finite type, Acta Appl. Math. 126 (2013) 35–63.
[4] Alexis Ballier, Maya Stein, The domino problem on groups of polynomial growth, arXiv:1311.4222, 2013.
[5] Robert Berger, The Undecidability of the Domino Problem, American Mathematical Society, 1966.
[6] William W. Boone, The word problem, Proc. Natl. Acad. Sci. USA 44 (1958) 1061–1065.
[7] Ceccherini-Silberstein Tullio, Michel Coornaert, Cellular Automata and Groups, Springer, 2009.
[8] Aubrey da Cunha, Turing machines on Cayley graphs, in: Lev D. Beklemishev, Ruy de Queiroz (Eds.), Logic, Language, Information and Computation, in: Lecture Notes in Computer Science, vol. 6642, Springer, Berlin Heidelberg, 2011, pp. 84–94.
[9] François Dahmani, private communication.
[10] François Dahmani, Asli Yaman, Symbolic dynamics and relatively hyperbolic groups, Groups Geom. Dyn. 2 (2) (2008) 165–184.
[11] Bruno Durand, Andrei E. Romashchenko, Alexander Shen, Effective closed subshifts in 1D can be implemented in 2D, in: Fields of Logic and Computation, 2010, pp. 208–226.
[12] Anahí Gajardo, Jacques Mazoyer, One head machines from a symbolic approach, Theoret. Comput. Sci. 370 (1–3) (2007) 34–47.
[13] Rostislav Grigorchuk, Degrees of growth of finitely generated groups, and the theory of invariant means, Math. USSR, Izv. 48 (5) (1984).
[14] Gustav Hedlund, Marston Morse, Symbolic dynamics, Amer. J. Math. 60 (4) (1938) 815–866.
[15] Mike Hochman, On the dynamics and recursive properties of multidimensional symbolic systems, Invent. Math. 176 (1) (2009) 131–167.
[16] Mike Hochman, Tom Meyerovitch, A characterization of the entropies of multidimensional shifts of finite type, Ann. of Math. 171 (3) (2010) 2011–2038.
[17] Douglas A. Lind, Brian Marcus, An Introduction to Symbolic Dynamics and Coding, Cambridge University Press, 1995.
[18] Roger C. Lyndon, Paul E. Schupp, Combinatorial Group Theory, Springer-Verlag, 1977.
[19] Pyotr Novikov, On the algorithmic unsolvability of the word problem in group theory, Proc. Steklov Inst. Math. 44 (1955) 143 (in Russian).
[20] Ville Salo, Ilkka Törmä, Group-Walking Automata, in: Cellular Automata and Discrete Complex Systems: 21st IFIP WG 1.5 International Workshop, AUTOMATA 2015, Proceedings, Turku, Finland, June 8–10, 2015, 2015, pp. 224–237.
[21] R.M. Robinson, Undecidable tiling problems in the hyperbolic plane, Invent. Math. 44 (1978) 159–264.
[22] Raphael Robinson, Undecidability and nonperiodicity for tilings of the plane, Invent. Math. 12 (1971) 177–209.
[23] John R. Stallings, On torsion-free groups with infinitely many ends, Ann. of Math. 88 (2) (1968) 312–334.
[24] Asli Yaman, private communication.